Extending the Description Horn Logic DHL

Linh Anh Nguyen

Institute of Informatics
University of Warsaw

CS&P'09

Kraków-Przegorzały, September 2009

Outline

Description Logics

- Description logics (DLs) are a family of knowledge representation languages which can be used to represent the terminological knowledge of an application domain in a structured and formally well-understood way.
- DLs describe domain in terms of concepts (classes), roles (relationships) and individuals.
- DLs, as decidable fragments of FOL, have well-defined formal semantics.
- DLs are used as the main formal background for a number of languages used in the semantic web technology, including ontology engineering, reasoning with ontology-based markup (meta-data), service description and discovery.

Motivations

Problem

- The data complexity of the DL $\mathcal{ALC}$ is NP-complete.
- It is worth studying formalisms with PTIME data complexity.

Horn Fragments of Logic

- Horn rules are widely used in knowledge representation.
- Horn fragments usually have
  - a lower complexity or data complexity
  - efficient computational methods.

Approaches

- intersection of a DL with the Horn fragment of FOL
- combination of Horn fragments of a DL and FOL

DL Architecture

- **Knowledge base**
  - **RBox** (axioms about roles)

    $$hasChild \sqsubseteq hasDescendant$$
    $$hasDescendant \circ hasDescendant \sqsubseteq hasDescendant$$
    $$hasParent = hasChild^{-}$$

  - **TBox** (definitions of concepts and terminological axioms)

    $$Parent = \exists hasChild.\top$$
    $$Father = Parent \sqcap Male$$
    $$Mother = Parent \sqcap Female$$

  - **ABox** (assertions about instances)

    $$John : Father$$
    $$Mary : Mother$$
    $$hasChild(John, Jack)$$

- Inference system
- Interface

Description Logic $\mathcal{SHI}$ (1)

Syntax	Example	Semantics w.r.t. $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$
a	*John*	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
A	*Human*	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
r	*hasChild*	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
r^-	$hasChild^-$	$(r^-)^{\mathcal{I}} = \{(x, y) \mid (y, x) \in r^{\mathcal{I}}\}$
We use letters like R, S to denote a role of the form r or r^-.		
$C \sqcap D$	$Human \sqcap Male$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$Mother \sqcup Father$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\neg C$	$\neg Male$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$\exists R.C$	$\exists hasChild.Human$	$\{x \mid \exists y.(x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
$\forall R.C$	$\forall hasChild.Doctor$	$\{x \mid \forall y.(x, y) \in R^{\mathcal{I}} \to y \in C^{\mathcal{I}}\}$

Description Logic $\mathcal{SHI}$ (2)

RBox (axioms about roles)

Syntax	**Example**	**Semantics w.r.t. $\mathcal{I}$**
$R \sqsubseteq S$	$\mathit{hasChild} \sqsubseteq \mathit{hasDescendant}$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
$R \circ R \sqsubseteq R$	...	$R^{\mathcal{I}}$ is transitive

TBox (terminological axioms)

Syntax	**Example**	**Semantics w.r.t. $\mathcal{I}$**
$C \sqsubseteq D$	$\mathit{Mother} \sqsubseteq \mathit{Parent}$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$C = D$	$\mathit{Parent} = \mathit{Mother} \sqcup \mathit{Father}$	$C^{\mathcal{I}} = D^{\mathcal{I}}$

ABox (assertions)

Syntax	**Example**	**Semantics w.r.t. $\mathcal{I}$**
$a : A$	$\mathit{John} : \mathit{Father}$	$a^{\mathcal{I}} \in A^{\mathcal{I}}$
$R(a, b)$	$\mathit{hasChild}(\mathit{John}, \mathit{Jack})$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

The Description Horn Logic DHL

DHL was introduced by Grosof et al. as a fragment of DL $\mathcal{SHI}$ with the following restriction:

- a TBox is a finite set of axioms of the form

$$C_b \sqsubseteq C_h \ \text{ or } \ \top \sqsubseteq \forall R.C_h$$

  where C_b and C_h are defined by the following BNF grammar:

$$C_h ::= A \mid C_h \sqcap C_h \mid \forall R.C_h$$

$$C_b ::= A \mid C_b \sqcap C_b \mid C_b \sqcup C_b \mid \exists R.\top \mid \exists R.C_b$$

Data Complexity of DHL

The Instance Checking Problem in DHL

Is an individual a an **instance** of a concept C_b w.r.t. a DHL knowledge base $(\mathcal{R}, \mathcal{T}, \mathcal{A})$?

- i.e., is $a^{\mathcal{I}} \in C_b^{\mathcal{I}}$ for all model $\mathcal{I}$ of $(\mathcal{R}, \mathcal{T}, \mathcal{A})$?

Data Complexity

The data complexity of the instance checking problem is measured w.r.t. to the size of $\mathcal{A}$, assuming that $\mathcal{R}$, $\mathcal{T}$, C_b and a are fixed.

Theorem (Grosof et al)

The instance checking problem in DHL has PTIME data complexity.

Context-Free Description Logic with Inverse Roles $\mathcal{ALCI}_{cf}$

$\mathcal{ALCI}_{cf}$ extends $\mathcal{SHI}$ by allowing role axioms of the form

$$R_1 \circ \ldots \circ R_k \sqsubseteq S$$

where $k \geq 0$ and the l.h.s. stands for the identity relation if $k = 0$.

The semantics of such an axiom w.r.t. an interpretation $\mathcal{I}$ is that

$$R_1^{\mathcal{I}} \circ \ldots \circ R_k^{\mathcal{I}} \subseteq S^{\mathcal{I}}$$

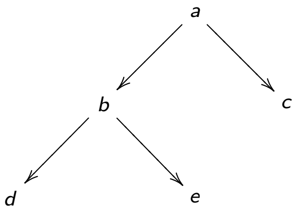
The Extended Description Horn Logic EDHL

EDHL extends DHL by allowing role axioms of the form

$$R_1 \circ \ldots \circ R_k \sqsubseteq S$$

EDHL: Example (1)

Consider the binary tree



specified by the following ABox

$$\mathcal{A} = \{root(a), L(a,b), R(a,c), L(b,d), R(b,e), middle\_level(b)\}$$

where

- $L(x,y)$ stands for "y is the left successor of x"
- $R(x,y)$ stands for "y is the right successor of x".

EDHL: Example (2)

Let $\mathcal{R}$ be the following RBox:

$$L \sqsubseteq S \qquad id \sqsubseteq T$$
$$R \sqsubseteq S \qquad S^- \circ T \circ S \sqsubseteq T$$

This RBox defines S and T to be the roles such that

- $S(x, y)$ means y is a successor of x
- $T(x, y)$ means x and y are at the same level of the tree.

Note: $\mathcal{R}$ is not a "regular" RBox.

EDHL: Example (3)

Let $\mathcal{T}$ be the TBox consisting of the following axioms:

$$\begin{aligned}
\exists T.\exists S.\top &\sqsubseteq inner_level \\
middle_level &\sqsubseteq \forall T.middle_level \\
root &\sqsubseteq on_left_path \\
\exists L^{-}.on_left_path &\sqsubseteq on_left_path
\end{aligned}$$

which means

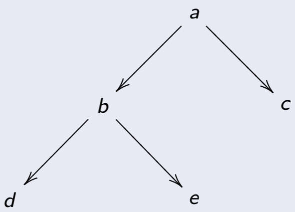
$$T(x,y) \land S(y,z) \rightarrow inner\_level(x)$$
$$middle\_level(x) \land T(x,y) \rightarrow middle\_level(y)$$
$$root(x) \rightarrow on\_left\_path(x)$$
$$L(y,x) \land on\_left\_path(y) \rightarrow on\_left\_path(x)$$

EDHL: Example (4)

$L \sqsubseteq S$	$id \sqsubseteq T$
$R \sqsubseteq S$	$S^{-} \circ T \circ S \sqsubseteq T$

$$\exists T.\exists S.\top \sqsubseteq inner\_level$$
$$middle\_level \sqsubseteq \forall T.middle\_level$$
$$root \sqsubseteq on\_left\_path$$
$$\exists L^{-}.on\_left\_path \sqsubseteq on\_left\_path$$

$root(a), L(a,b), R(a,c), L(b,d), R(b,e), middle\_level(b)$



It can be shown that

- c is an instance of concept $middle\_level$
- a, b and d are instances of concept on_left_path

EDHL: Properties

Datalog

- A **Datalog knowledge base** consists of
  - **extensional part**: a set of facts (ground atoms) in FOL
  - **intensional part**: a Datalog program, i.e. a logic program in FOL without negation and function symbols, satisfying the "range-restrictedness" condition.
- A **query** to a Datalog knowledge base is a conjunction of (relational) atoms. **Answers** for a query are defined as usual.

Theorem

- The instance checking problem in EDHL is reducible to the query answering problem in Datalog.
- The instance checking problem in EDHL has PTIME data complexity.

CARIN Approach (Levy & Rousset)

In the CARIN approach, a knowledge base consists of:

- **the bottom layer**: a terminology
  for defining concepts and roles
- **the top layer**: a logic program of FOL
  for defining other predicates.

Datalog-Like Knowledge Bases Using EDHL

EDHL-Datalog

- A **knowledge base** $(\mathcal{R}, \mathcal{T}, \mathcal{P}, \mathcal{A})$ in EDHL-Datalog consists of:
    - an RBox $\mathcal{R}$
    - an EDHL TBox $\mathcal{T}$
    - a Datalog program $\mathcal{P}$, which may use concept names as unary predicates and role names as binary predicates in bodies of program clauses
    - a set $\mathcal{A}$ of ground facts (i.e. ground atomic formulas).
- A **query** to a knowledge base in EDHL-Datalog is a conjunction of (relational) atoms, as in the case of Datalog.
- **Answers** to a query are defined as usual.

EDHL-Datalog: Example

$$\mathcal{R} = \emptyset$$

$$\begin{aligned}\mathcal{T} = \{&\exists hasChild.\top \sqsubseteq parent,\\ &parent \sqcap male \sqsubseteq father,\\ &parent \sqcap female \sqsubseteq mother\}\end{aligned}$$

$$\begin{aligned}\mathcal{P} = \{&father(x) \land hasChild(x,y) \land age(y,k) \land k \leq 3 \rightarrow discount(x,10),\\ &mother(x) \land hasChild(x,y) \land age(y,k) \land k \leq 3 \rightarrow discount(x,15)\}\end{aligned}$$

$$\begin{aligned}\mathcal{A} = \{&female(Jane), male(Mike), male(Peter),\\ &hasChild(Jane, Peter), hasChild(Mike, Peter), age(Peter, 2)\}\end{aligned}$$

where $\leq$ is a special predicate with the usual semantics.

EDHL-Datalog: Properties

Theorem

- A knowledge base in EDHL-Datalog can effectively be translated into an equivalent knowledge base in Datalog.
- The data complexity of EDHL-Datalog is in $\mathrm{PT{\scriptsize IME}}$.

GDHL: A Further Extension with Function Symbols

GDHL extends EDHL-Datalog by:

- **allowing the constructor $\exists R.C$ to appear in the r.h.s. of terminological inclusion axioms $C_b \sqsubseteq C_h$**
  - for example:

    $parent \sqsubseteq \exists hasChild.\top$

    $\exists hasChild.\top \sqsubseteq parent$
- **using definite logic programs (of FOL) instead of Datalog programs**
  - for example:

    $father(x) \land hasChild(x, y) \land age(y, k) \land leq(k, s^3(0)) \rightarrow discount(x, 10)$

GDHL: Definition

A **GDHL TBox** is a finite set of axioms of the form $C_b \sqsubseteq C_h$, where C_b and C_h are defined by the following BNF grammar:

$$
\begin{aligned}
C_h &::= A \mid C_h \sqcap C_h \mid \exists R.\top \mid \exists R.C \mid \forall R.C_h \\
C_b &::= \top \mid A \mid C_b \sqcap C_b \mid C_b \sqcup C_b \mid \exists R.C_b
\end{aligned}
$$

A **knowledge base in GDHL** is a tuple $(\mathcal{R}, \mathcal{T}, \mathcal{P})$, where

- $\mathcal{R}$ is an RBox
- $\mathcal{T}$ is a GDHL TBox
- $\mathcal{P}$ is a definite logic program, which may use concept names as unary predicates and role names as binary predicates (only) in the bodies of its program clauses.

GDHL: Properties

Proposition

A knowledge base in GDHL can effectively be translated into an equivalent definite logic program.

Conclusions

- We have formulated the useful extensions EDHL, EDHL-Datalog and GDHL of DHL.
  - Each of these extensions is more expressive than the previous.
  - None of them was studied before.
- The instance checking problem in EDHL has PTIME data complexity.
- The query language EDHL-Datalog has PTIME data complexity.
- The translation from EDHL-Datalog into Datalog allows using efficient computational methods of Datalog.
- EDHL-Datalog is more convenient than Datalog for Semantic Web.
- Answering a query to a knowledge base in GDHL is reducible to answering a query to a definite logic program, for which advanced methods can be used.

Future Work

We intend to extend EDHL and EDHL-Datalog by

- allowing number restrictions and negation to appear in the left hand side of terminological inclusion axioms
- allowing negation to appear in bodies of Datalog program clauses
- using stratified model semantics or well-founded semantics to deal with negation.