Rewriting Conjunctive Queries under DL Constraints

Héctor Pérez-Urbina, Boris Motik, and Ian Horrocks

Computing Laboratory
University of Oxford

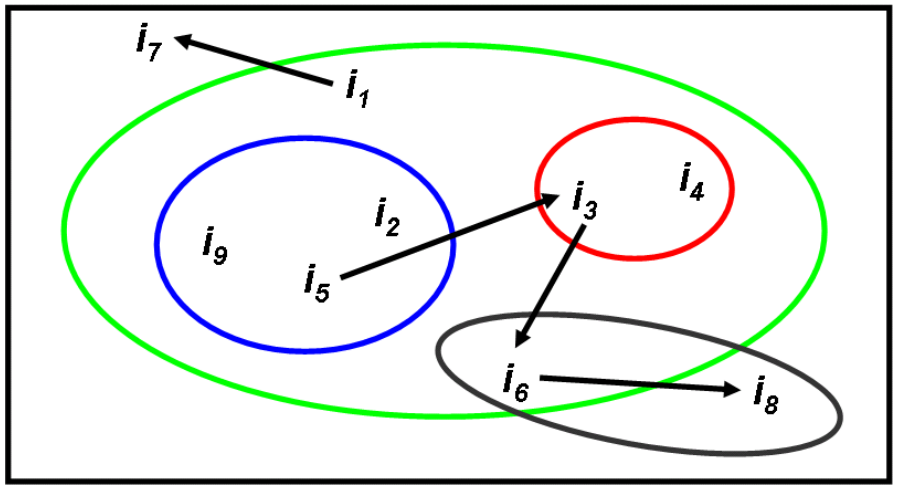
International Workshop on
Logic in Databases
May 2008

Outline

- Introduction
  - Description Logics (DLs)
  - DLs and DBs
- Answering queries via query rewriting
- Query rewriting for $\mathcal{ELHI}$
- Complexity results
- Conclusion
- Future work

The Description Logic World

- Things (individuals), relationships between things (roles) and sets of things (concepts)



A DL Knowledge Base (KB) is a tuple $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

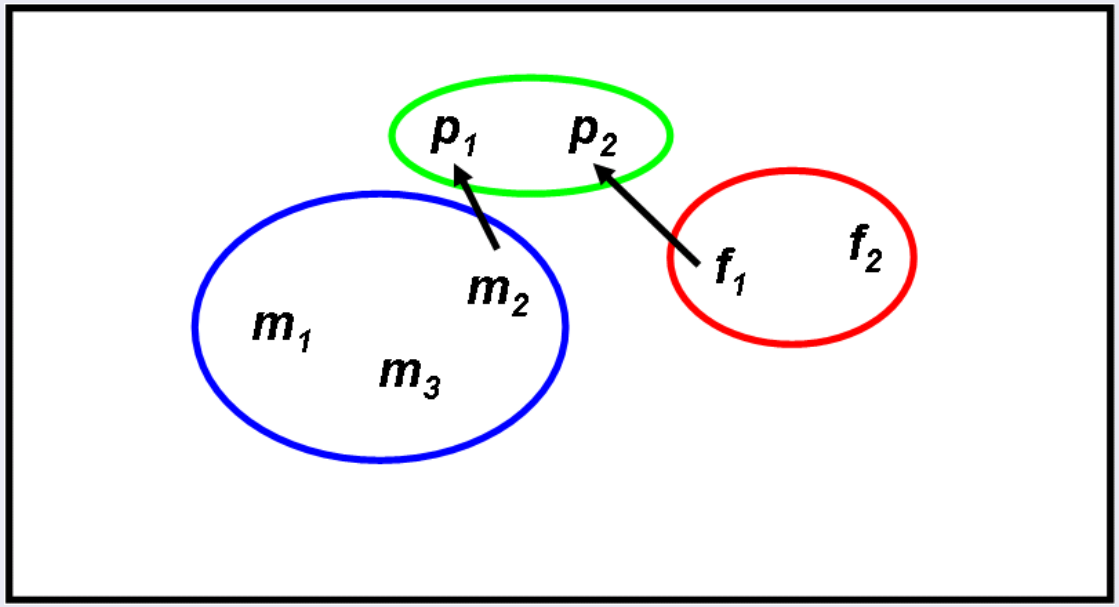
- TBox $\approx$ Conceptual schema
- ABox $\approx$ (Partial) Database instance

DL Knowledge Bases

Example

$\mathcal{T} = \{\text{Parent} \sqsubseteq \exists \text{hasChild}, \text{Mother} \sqsubseteq \text{Parent}, \text{Father} \sqsubseteq \text{Parent}\}$

$\mathcal{A} = \{\text{Parent}(p_1), \text{Parent}(p_2), \text{Father}(f_1), \text{Father}(f_2), \text{Mother}(m_1),$
$\text{Mother}(m_2), \text{Mother}(m_3), \text{hasChild}(m_2, p_1), \text{hasChild}(f_1, p_2)\}$



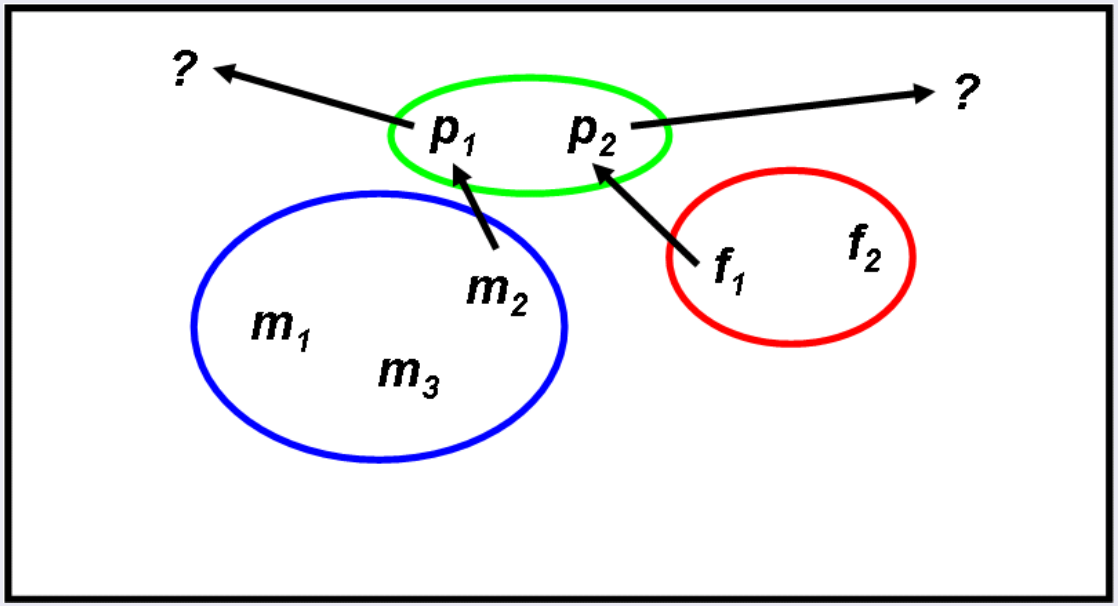
- Parent
- Mother
- Father

DL Knowledge Bases

Example

$\mathcal{T} = \{\text{Parent} \sqsubseteq \exists \text{hasChild}, \text{Mother} \sqsubseteq \text{Parent}, \text{Father} \sqsubseteq \text{Parent}\}$

$\mathcal{A} = \{\text{Parent}(p_1), \text{Parent}(p_2), \text{Father}(f_1), \text{Father}(f_2), \text{Mother}(m_1),$
$\text{Mother}(m_2), \text{Mother}(m_3), \text{hasChild}(m_2, p_1), \text{hasChild}(f_1, p_2)\}$



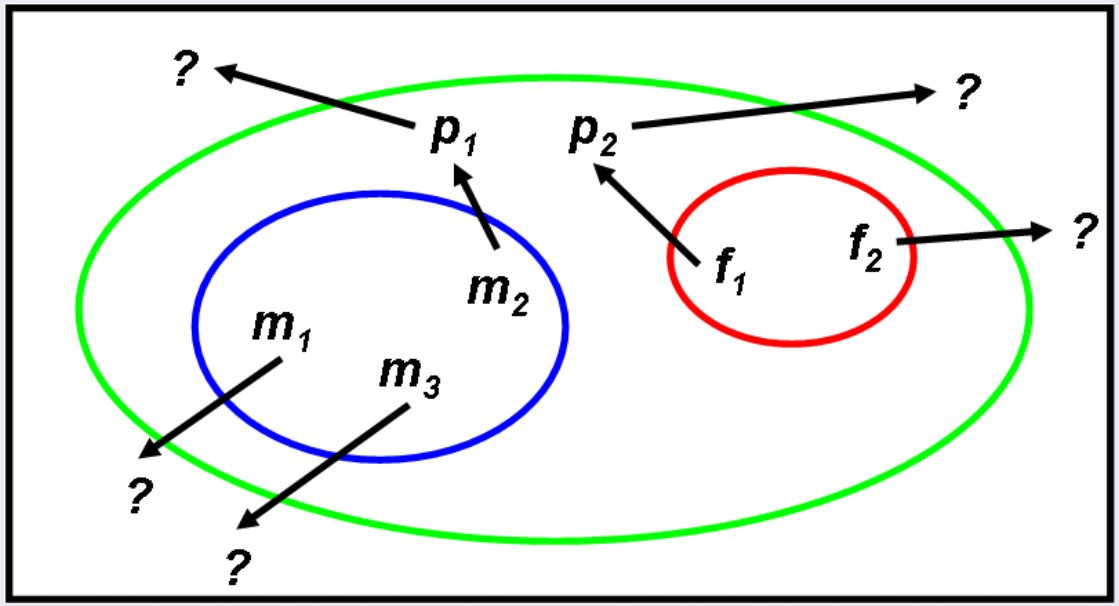
- Parent
- Mother
- Father

DL Knowledge Bases

Example

$\mathcal{T} = \{\text{Parent} \sqsubseteq \exists \text{hasChild}, \text{Mother} \sqsubseteq \text{Parent}, \text{Father} \sqsubseteq \text{Parent}\}$

$\mathcal{A} = \{\text{Parent}(p_1), \text{Parent}(p_2), \text{Father}(f_1), \text{Father}(f_2), \text{Mother}(m_1),$
$\text{Mother}(m_2), \text{Mother}(m_3), \text{hasChild}(m_2, p_1), \text{hasChild}(f_1, p_2)\}$



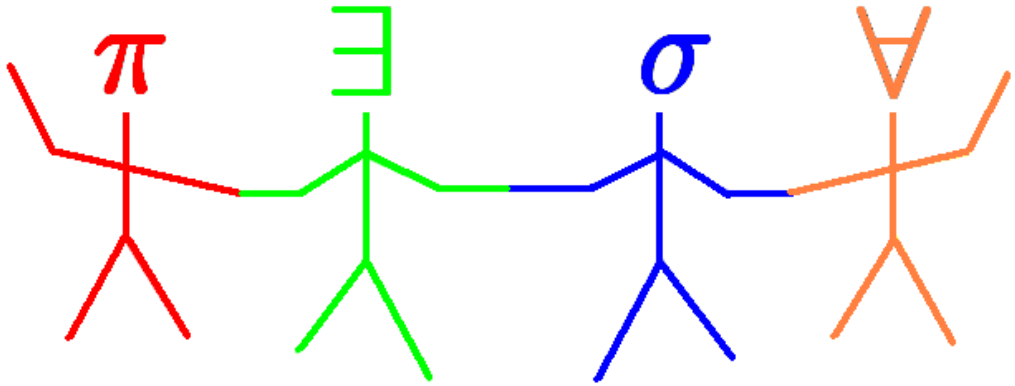
- Parent
- Mother
- Father

Description Logic Applications

- DLs underpin the W3C standard OWL ontology languages
- Conceptual modeling
  - Systematized Nomenclature of Medicine (Snomed)
  - National Cancer Institute (NCI)
  - Generalized Architecture for Languages, Encyclopaedias and Nomenclatures in medicine (Galen)
- Semantic Web
- Information Integration
- Data exchange
- Data warehousing

Description Logics and Databases

- TBox $\approx$ Conceptual schema (set of dependencies)
- ABox $\approx$ (Partial) Database instance
- Query answering (QA) over DL KBs $\equiv$ QA over incomplete DBs
  - First-order entailment!
  - $\text{ans}(\phi, \mathcal{K}) = \{\vec{a} \mid \mathcal{K} \models \phi[a_1/x_1, \ldots, a_n/x_n]\}$



Querying a Knowledge Base

Example

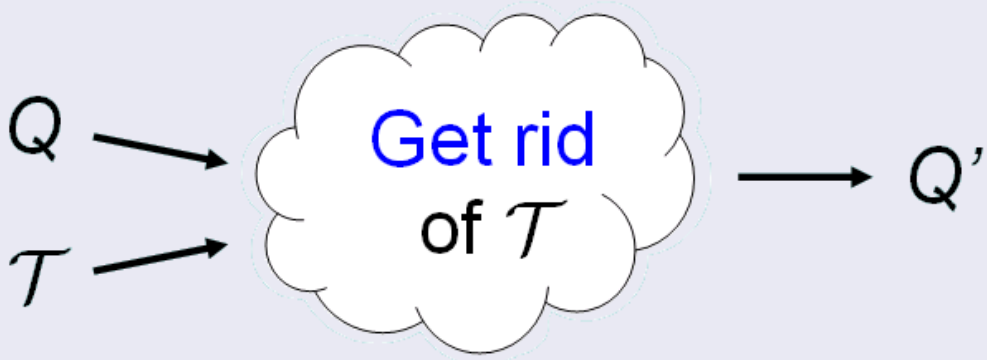
$\mathcal{T} = \{\exists \text{hasParent.Human} \sqsubseteq \text{Human}, \text{Child} \sqsubseteq \exists \text{hasParent.Human},$
$\text{Father} \sqsubseteq \text{Human}, \text{hasFather} \sqsubseteq \text{hasParent}\}$

$\mathcal{A} = \{\text{Father}(f_1), \text{Father}(f_2), \text{Child}(c_1), \text{hasFather}(s_1, f_2)\}$

- Q = Human(x)
- $\mathcal{T} \models$
  - Father $\sqsubseteq$ Human
  - Child $\sqsubseteq$ Human
  - $\exists$hasFather.Human $\sqsubseteq$ Human
- $\text{ans}(Q, \mathcal{K}) = \{\langle f_1 \rangle, \langle f_2 \rangle, \langle c_1 \rangle, \langle s_1 \rangle\}$
- Consider the TBox first!

Query Answering via Query Rewriting

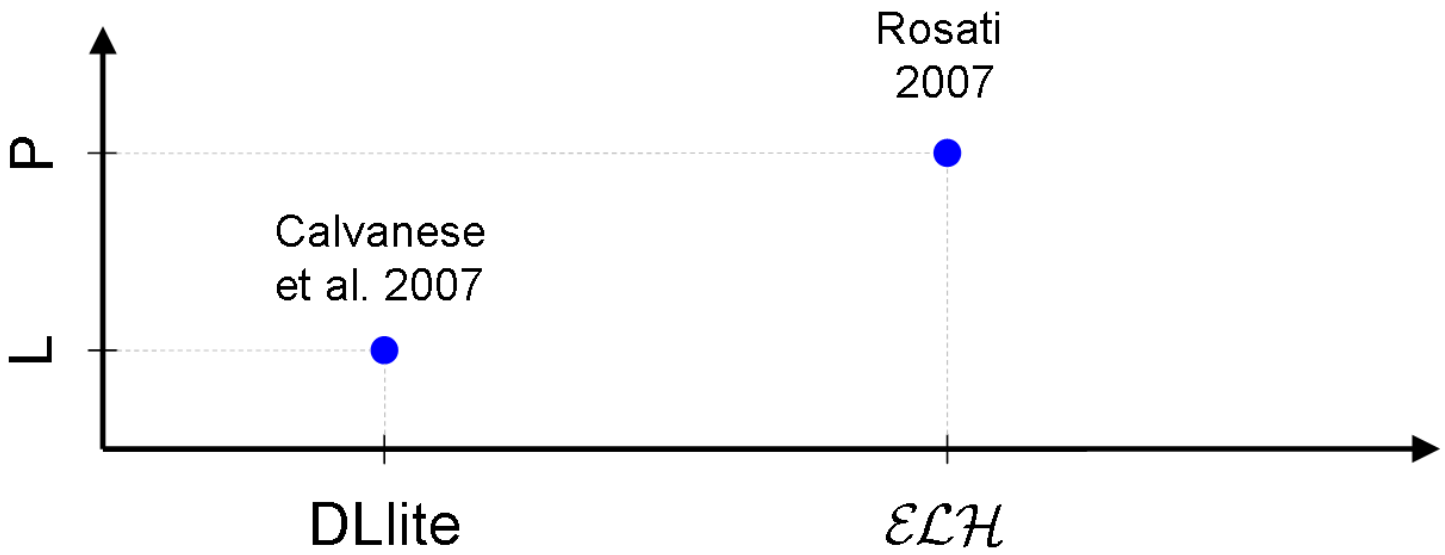
Query Rewriting over DL TBoxes



s.t. $\text{ans}(Q, \langle \mathcal{T}, \mathcal{A} \rangle) = \text{ans}(Q', \langle \emptyset, \mathcal{A} \rangle)$, for all $\mathcal{A}$

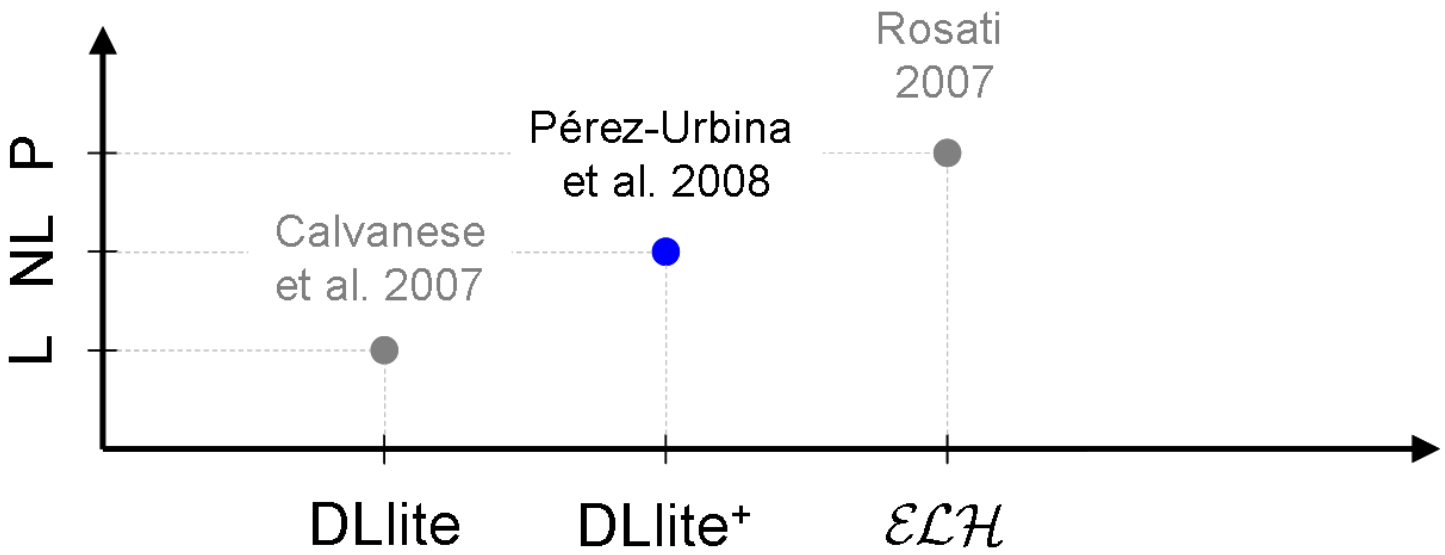
- Compile the knowledge of $\mathcal{T}$ into Q
  - Q = Human(x)
  - Q' = Human(x) $\vee$ Child(x) $\vee$ Father(x) $\vee \ldots$
- Avoid repeating work with different ABoxes

Related Work



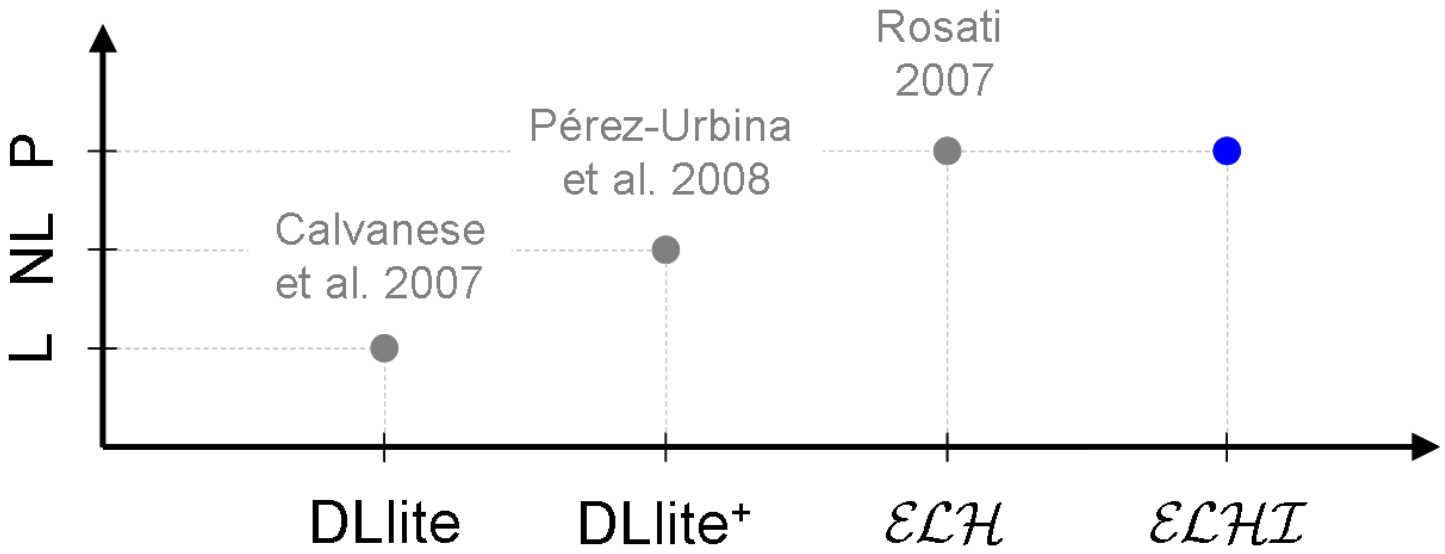
- Type of Rewriting
  - DL-Lite $\mapsto$ Union of CQs
  - $\mathcal{ELH} \mapsto$ Datalog program

Related Work



- Type of Rewriting
  - DL-Lite $\mapsto$ Union of CQs
  - DL-Lite$^+$ $\mapsto$ Linear datalog program
  - $\mathcal{ELH} \mapsto$ Datalog program

Our Goal



- Generalization and extension of existing approaches
- Worst-case optimal
  - DL-Lite $\mapsto$ Union of CQs
  - DL-Lite$^+$ $\mapsto$ Linear datalog program
  - $\mathcal{ELH}, \mathcal{ELHI} \mapsto$ Datalog program

Querying $\mathcal{ELHI}$ KBs

$\mathcal{ELHI}$

- $B \rightarrow A \mid \exists R \mid \exists R.A \mid B_1 \sqcap B_2$
- $R \rightarrow P \mid P^-$
- TBox assertions: $B_1 \sqsubseteq B_2$, and $R_1 \sqsubseteq R_2$
- ABox assertions: $A(a)$, and $P(a, b)$

Queries

- $Q = \langle Q_P, P \rangle$
- P is a set of Horn clauses

Using Resolution

- Goal: Worst-case optimal query rewriting algorithm for $\mathcal{ELHI}$ TBoxes
- Idea: Use known approaches
  - Motik and Kazakov: resolution!

Example

$\mathcal{T} = \{\text{Human} \sqsubseteq \exists \text{hasParent.Human}\}$

$\Xi(\mathcal{T}) = \{\mathit{hasParent}(x, f(x)) \leftarrow \mathit{Human}(x), \mathit{Human}(f(x)) \leftarrow \mathit{Human}(x)\}$

- Functional terms + cycles = trouble!

Our Algorithm in a Nutshell

Given a conjunctive query $Q = \langle Q_P, P \rangle$ and a $\mathcal{ELHI}$ TBox $\mathcal{T}$

- 1. Derive enough consequences from Q and $\mathcal{T}$
- 2. Get rid of functional symbols
- 3. Optimize by unfolding

1. Saturation

- Resolution with free selection calculus $\mathcal{R}^{DL}$
- Saturate $\Xi(\mathcal{T})$ and P to derive new consequences

Example

$Q = \langle Q_P, \{Q_P(x) \leftarrow Human(x)\}\rangle$
$\mathcal{T} = \{\exists \text{hasParent}^{-}.\text{Human} \sqsubseteq \text{Human}, \text{Parent} \sqsubseteq \exists \text{hasParent}^{-}.\text{Human}\}$

(1) $Human(x) \leftarrow \underline{hasParent(y,x)} \wedge Human(y)$
(2) $\underline{hasParent(g(x),x)} \leftarrow Parent(x)$
(3) $\underline{Human(g(x))} \leftarrow Parent(x)$
(4) $Q_P(x) \leftarrow \underline{Human(x)}$

---

(5) $Human(x) \leftarrow Parent(x) \wedge \underline{Human(g(x))}$ $(1,2)$
(6) $Human(x) \leftarrow \underline{Parent(x)}$ $(3,5)$

...

- Main challege: ensure termination!

2. Elimination of functional terms

ff$(Q, \mathcal{T})$: all function-free clauses in the saturation of $\Xi(\mathcal{T}) \cup P$

Example Contd.

$$\text{ff}(Q, \mathcal{T}) = \{Human(x) \leftarrow hasParent(y, x) \wedge Human(y),$$
$$Human(x) \leftarrow Parent(x), Q_P(x) \leftarrow Human(x)\}$$

Lemma

Given a conjunctive query Q and a $\mathcal{ELHI}$ TBox $\mathcal{T}$,
ans$(Q, \langle \mathcal{T}, \mathcal{A} \rangle) =$ ans(ff$(Q, \mathcal{T}), \langle \emptyset, \mathcal{A} \rangle)$, for all $\mathcal{A}$

- ff$(Q, \mathcal{T}$ is a rewriting!
- but is it optimal?

What happens with DL-Lite$^+$?

Example

$Q = \langle Q_P, \{Q_P(x) \leftarrow \mathit{Human}(x)\}\rangle$
$\mathcal{T} = \{\exists \mathsf{hasParent}.\mathsf{Human} \sqsubseteq \mathsf{Human}, \mathsf{hasFather} \sqsubseteq \mathsf{hasParent}\}$

$\mathrm{ff}(Q, \mathcal{T}) = \{\mathit{Human}(x) \leftarrow \mathit{hasParent}(y, x) \wedge \mathit{Human}(y),$
$\mathit{hasParent}(x, y) \leftarrow \mathit{hasFather}(x, y), Q_P(x) \leftarrow \mathit{Human}(x)\}$

- We expect a linear datalog program!

From Nonlinear to Linear datalog programs

(1) $\mathit{Human}(x) \leftarrow \mathit{hasParent}(y, x) \wedge \mathit{Human}(y)$
(2) $\mathit{hasParent}(x, y) \leftarrow \mathit{hasFather}(x, y)$

---

(1) $\mathit{Human}(x) \leftarrow \mathit{hasParent}(y, x) \wedge \mathit{Human}(y)$
(2) $\mathit{Human}(x) \leftarrow \mathit{hasFather}(y, x) \wedge \mathit{Human}(y)$

3. Unfolding

rew$(Q, \mathcal{T})$: unfold all clauses of the forms

- $A(x) \leftarrow B(x)$,
- $A(x) \leftarrow R(x, y)$,
- $A(x) \leftarrow R(y, x)$,
- $S(x, y) \leftarrow R(x, y)$, and
- $S(x, y) \leftarrow R(y, x)$

in ff$(Q, \mathcal{T})$, and get rid of them

Example Contd.

$$\begin{aligned}\text{rew}(Q, \mathcal{T}) = \{&\mathit{Human}(x) \leftarrow \mathit{hasParent}(y, x) \wedge \mathit{Human}(y),\\ &\mathit{Human}(x) \leftarrow \mathit{hasFather}(y, x) \wedge \mathit{Human}(y),\\ &Q_P(x) \leftarrow \mathit{Human}(x)\}\end{aligned}$$

Our Algorithm in a Nutshell

Phases

Given a conjunctive query Q and a $\mathcal{ELHI}$ TBox $\mathcal{T}$

- 1. Saturate to get consequences from Q and $\mathcal{T}$ using $\mathcal{R}^{DL}$
- 2. Get rid of functional symbols: $\mathsf{ff}(Q, \mathcal{T})$
- 3. Optimize by unfolding: $\mathsf{rew}(Q, \mathcal{T})$

Theorem

Given a conjunctive query Q and a $\mathcal{ELHI}$ TBox $\mathcal{T}$,
$\mathsf{ans}(Q, \langle \mathcal{T}, \mathcal{A} \rangle) = \mathsf{ans}(\mathsf{rew}(Q, \mathcal{T}), \langle \emptyset, \mathcal{A} \rangle)$, for all $\mathcal{A}$

Properties of the Rewriting

Lemma

If $\mathcal{T}$ is in $\mathcal{ELHI}$ or $\mathcal{ELH}$, then rew$(Q, \mathcal{T})$ is a datalog program

Lemma

Let rew$(Q, \mathcal{T}) = \langle Q_P, P' \rangle$. If $\mathcal{T}$ is in DL-Lite$^+$, then $P' = U_Q \cup U_C$, such that $\langle Q_P, U_Q \rangle$ is a union of conjunctive queries and $\langle Q_P, U_C \rangle$ is a linear datalog query

Lemma

If $\mathcal{T}$ is in DL-Lite, then rew$(Q, \mathcal{T})$ is a union of conjunctive queries

Complexity Analysis

Theorem

For a conjunctive query Q and a $\mathcal{ELHI}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, deciding whether $\vec{a} \in \text{ans}(Q, \mathcal{K})$ is P-complete data complexity

Corollary

Given a conjunctive query Q and a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, if $\mathcal{T}$ is in $\mathcal{ELH}$, DL-Lite$^+$, or DL-Lite, we can compute the answers to $\text{rew}(Q, \mathcal{T})$ over $\mathcal{A}$ in P, NL, or L respectively w.r.t. data complexity

- $\text{rew}(Q, \mathcal{T})$ is optimal!

Conclusion

New result

QA over $\mathcal{ELHI}$ KBs is P-complete data complexity

Contribution

Conjunctive query rewriting algorithm for $\mathcal{ELHI}$ KBs

- Worst-case optimal for sublanguages of $\mathcal{ELHI}$ for which QA is P-complete, NL-complete, or in L data complexity
- Generalization and extension of existing approaches
- Straightforward use of DB technology for query evaluation

Current and Future Work

Current Work

Nominals: $\mathcal{ELHIO}$

- $\{Pope\}$

Future Work

Prototype system in an Information Integration setting

- ComparaGRID

THANKS!

¿?