Optimising Resolution Based Rewriting Algorithms for DL Ontologies

Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos Stamou

National Technical University of Athens, Greece

Framework

we address the problem of *query rewriting* over expressive DL ontologies in the framework of *resolution*

- query rewriting is a key approach for query answering in the case of expressive DLs

$$\mathsf{ans}(\mathcal{Q} \cup \mathcal{T}, \mathcal{D}) = \mathsf{ans}(\mathcal{P}, \mathcal{D})$$

  - the datalog program $\mathcal{P}$ is a rewriting of the CQ $\mathcal{Q}$ over the TBox $\mathcal{T}$, for any dataset $\mathcal{D}$

Related work

rewriting-based systems:

- for DL-Lite,
  - QuOnto (Acciarri et al., 2005)
  - Presto (Rosati et al.,2010)
  - Rapid (Chortaras et al., 2011)
  - Quest (Rodriguez-Muro et al., 2012)
  - IQAROS (Venetis et al., 2013)
- for Datalog$_{\pm}$,
  - Nyaya (Gottlob et al., 2011)
- for more expressive DLs,
  - Requiem system (Perez-Urbina et al., 2009) for $\mathcal{ELHIO}$
  - Clipper (Eiter et al., 2012) for Horn-$\mathcal{SHIQ}$
  - KAON2 (Motik et al. 2005) for $\mathcal{SHIQ}$ and ground queries

Motivation

- resolution-based reasoning algorithms are worst case optimal
- the resolution technique allows for optimisations (ordering restrictions, subsumption deletion)

however,

- dead end paths, clauses with functional terms
- long inference paths
- subsumed clauses
- algorithms are unguided and exhaustive

scalability issues when the expressivity of the DL increases

Motivation Example 1

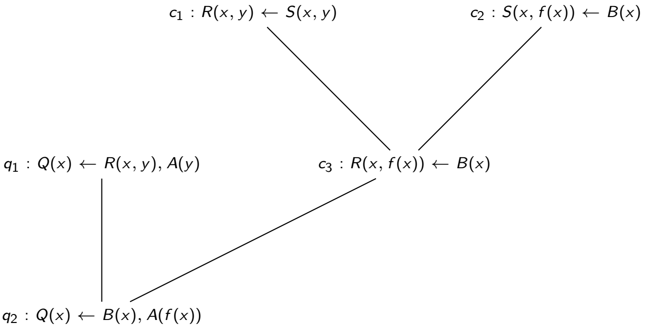
$$\begin{array}{lcl|l} S \sqsubseteq R & \rightsquigarrow & R(x,y) \leftarrow S(x,y) & q_1 : Q(x) \leftarrow R(x,y) \wedge A(y) \\ B \sqsubseteq \exists S & \rightsquigarrow & S(x,f(x)) \leftarrow B(x) & \end{array}$$

Motivation Example 1

$S \sqsubseteq R \quad \rightsquigarrow \quad R(x,y) \leftarrow S(x,y)$

$B \sqsubseteq \exists S \quad \rightsquigarrow \quad S(x,f(x)) \leftarrow B(x)$

$q_1 : Q(x) \leftarrow R(x,y) \wedge A(y)$



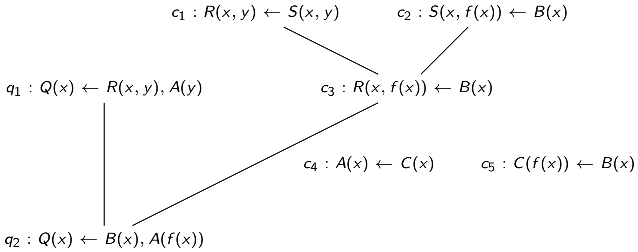
$\mathcal{R}_1 = \{q_1, c_1\}$

Description of work

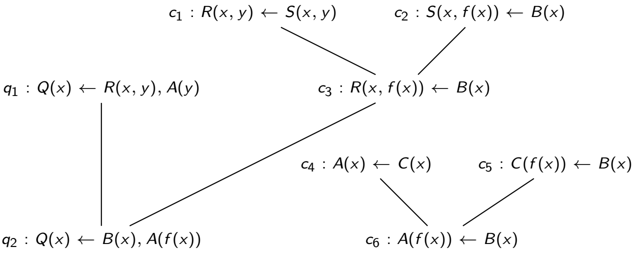
we propose a *goal oriented* resolution-based rewriting algorithm

- it supports $\mathcal{ELHI}$ (technically challenging)
- optimised resolution strategy
- circumvents the main drawbacks of resolution technique
- we conducted an experimental evaluation using large-scale real-world ontologies
- requires milliseconds for large scale ontologies which existing systems cannot handle

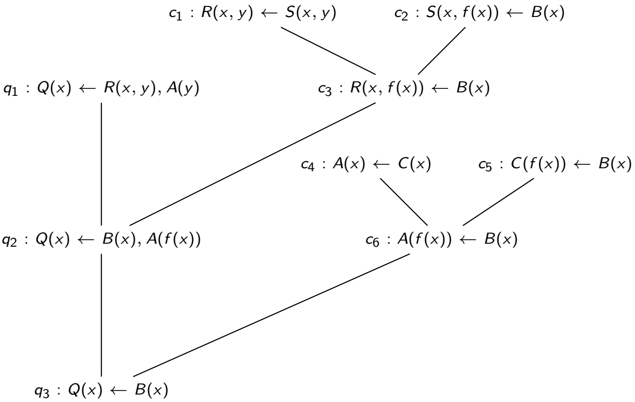
Resolution-based rewriting



Resolution-based rewriting

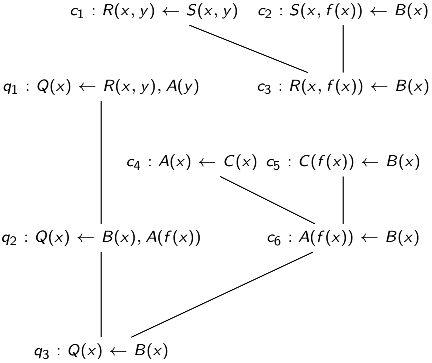


Resolution-based rewriting



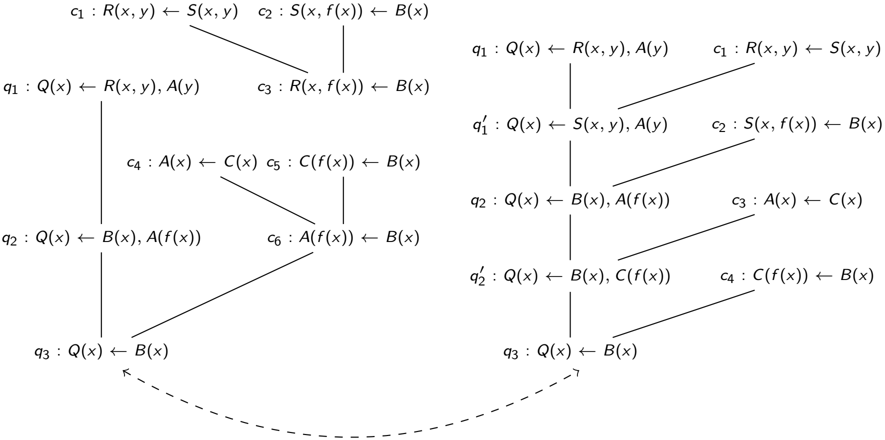
$$\mathcal{R} = \{c_1, q_1, c_4, q_3\}$$

Resolution-based rewriting



$\mathcal{R} = \{c_1, q_1, c_4, q_3\}$

Resolution-based rewriting



$\mathcal{R} = \{c_1, q_1, c_4, q_3\}$

$\mathcal{R}' = \mathcal{R} \cup \{q_1'\}$

Resolution-based rewriting

$q_1 : Q(x) \leftarrow R(x,y), A(y)$	$c_1 : R(x,y) \leftarrow S(x,y)$
$q_1' : Q(x) \leftarrow S(x,y), A(y)$	$c_2 : S(x, f(x)) \leftarrow B(x)$
$q_2 : Q(x) \leftarrow B(x), A(f(x))$	$c_3 : A(x) \leftarrow C(x)$
$q_2' : Q(x) \leftarrow B(x), C(f(x))$	$c_4 : C(f(x)) \leftarrow B(x)$
$q_3 : Q(x) \leftarrow B(x)$	

$$\mathcal{R}' = \mathcal{R} \cup \{q_1'\}$$

Resolution-based rewriting

$q_1 : Q(x) \leftarrow R(x,y), A(y)$ $\quad c_1 : R(x,y) \leftarrow S(x,y)$

$q'_1 : Q(x) \leftarrow S(x,y), A(y)$ $\quad c_2 : S(x,f(x)) \leftarrow B(x)$

$q_2 : Q(x) \leftarrow B(x), A(f(x))$ $\quad c_3 : A(x) \leftarrow C(x)$

$q'_2 : Q(x) \leftarrow B(x), C(f(x))$ $\quad c_4 : C(f(x)) \leftarrow B(x)$

$q_3 : Q(x) \leftarrow B(x)$

$\rightsquigarrow$

$q_1 : Q(x) \leftarrow R(x,y), A(y)$ $\quad c_1 : R(x,y) \leftarrow S(x,y)$

$q'_1 : Q(x) \leftarrow S(x,y), A(y)$ $\quad c_3 : A(x) \leftarrow C(x)$

$q^*_2 : Q(x) \leftarrow S(x,y), C(x)$ $\quad c_2 : S(x,f(x)) \leftarrow B(x)$

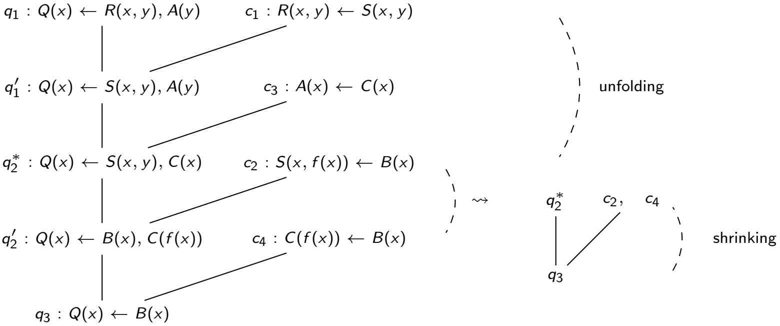
$q'_2 : Q(x) \leftarrow B(x), C(f(x))$ $\quad c_4 : C(f(x)) \leftarrow B(x)$

$q_3 : Q(x) \leftarrow B(x)$

$\mathcal{R}' = \mathcal{R} \cup \{q'_1\}$

$\mathcal{R}^* = \mathcal{R}' \cup \{q^*_2\}$

The calculus for DL-Lite



The calculus for DL-Lite

- *shrinking* rule,
  - hyper-resolution inference that produces directly clauses without any functional terms:
    - appropriate side premise clauses are selected in order to unify bound variables with functional terms which will be eliminated
    - resolvents differ from the main premise clause in that they do not contain one or more bound variables
- the *unfolding* rule,
  - does not affect the bound variables of the main premise

The case of $\mathcal{ELHI}$

- $\mathcal{ELHI}$ allows for clauses of the form: $A(x) \leftarrow R(x,y) \wedge C(y)$, $\mathcal{EL}$-clauses
- we cannot perform unfolding with $\mathcal{EL}$-clauses (variable proliferation$\rightsquigarrow$ termination issues)

$q_1 : Q(x) \leftarrow A(x)$ $\qquad$ $c_1 : A(x) \leftarrow R(x,y), C(y)$

$q_2 : Q(x) \leftarrow R(x,y), C(y)$

The case of $\mathcal{ELHI}$

- complex interaction between $\mathcal{EL}$-clauses

$$\underline{c_1 : K(x) \leftarrow R(y,x), A(y)} \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$c_3 : K(x) \leftarrow B(x), A(f(x))$$

The case of $\mathcal{ELHI}$

- complex interaction between $\mathcal{EL}$-clauses

$$c_1 : K(x) \leftarrow R(y,x), A(y) \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$c_3 : K(x) \leftarrow B(x), A(f(x))$$

$$c_4 : A(x) \leftarrow R(y,x), C(y) \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$c_5 : A(f(x)) \leftarrow B(x), C(x)$$

The case of $\mathcal{ELHI}$

- complex interaction between $\mathcal{EL}$-clauses

$$\underline{c_1 : K(x) \leftarrow R(y,x), A(y)} \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$\underline{c_4 : A(x) \leftarrow R(y,x), C(y)} \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$c_3 : K(x) \leftarrow B(x), A(f(x))$$

$$c_5 : A(f(x)) \leftarrow B(x), C(x)$$

$$c_6 : K(x) \leftarrow B(x), C(x)$$

The case of $\mathcal{ELHI}$

- complex interaction between $\mathcal{EL}$-clauses

$$c_1 : K(x) \leftarrow R(y,x), A(y) \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$c_3 : K(x) \leftarrow B(x), A(f(x))$$

$$c_4 : A(x) \leftarrow R(y,x), C(y) \qquad c_2 : R(f(x),x) \leftarrow B(x)$$

$$c_5 : A(f(x)) \leftarrow B(x), C(x)$$

$$c_6 : K(x) \leftarrow B(x), C(x)$$

clauses of the form c_5 and c_6 must participate in the rewriting process:

The case of $\mathcal{ELHI}$

- complex interaction between $\mathcal{EL}$-clauses

$c_1 : K(x) \leftarrow R(y,x), A(y)$

$c_2 : R(f(x),x) \leftarrow B(x)$

$c_3 : K(x) \leftarrow B(x), A(f(x))$

$c_4 : A(x) \leftarrow R(y,x), C(y)$

$c_2 : R(f(x),x) \leftarrow B(x)$

$c_5 : A(f(x)) \leftarrow B(x), C(x)$

$c_6 : K(x) \leftarrow B(x), C(x)$

clauses of the form c_5 and c_6 must participate in the rewriting process:

$q_1 : Q(x) \leftarrow K(x)$

$c_6 : K(x) \leftarrow B(x), C(x)$

$q_2 : Q(x) \leftarrow B(x), C(x)$

unfolding

The case of $\mathcal{ELHI}$

- complex interaction between $\mathcal{EL}$-clauses

$c_1 : K(x) \leftarrow R(y,x), A(y)$ $\qquad$ $c_2 : R(f(x),x) \leftarrow B(x)$

$c_3 : K(x) \leftarrow B(x), A(f(x))$

$c_4 : A(x) \leftarrow R(y,x), C(y)$ $\qquad$ $c_2 : R(f(x),x) \leftarrow B(x)$

$c_5 : A(f(x)) \leftarrow B(x), C(x)$

$c_6 : K(x) \leftarrow B(x), C(x)$

clauses of the form c_5 and c_6 must participate in the rewriting process:

unfolding:

$q_1 : Q(x) \leftarrow K(x)$ $\qquad$ $c_6 : K(x) \leftarrow B(x), C(x)$

$q_2 : Q(x) \leftarrow B(x), C(x)$

shrinking:

$q_2 : Q(x) \leftarrow R(x,y), A(y)$ $\qquad$ $c_2 : R(f(x),x) \leftarrow B(x)$, $c_5 : A(f(x)) \leftarrow B(x), C(x)$

$q_3 : Q(x) \leftarrow B(x), C(x)$

Extending the calculus for $\mathcal{ELHI}$

- we extend the DL-Lite calculus to obtain clauses of the form $A(f(x)) \leftarrow B(x), C(x)$
- we introduce the *function rule* that is applied on $\mathcal{EL}$-clauses:

$$c_4 : A(x) \leftarrow R(x, y), C(y) \qquad c_5 : R(f(x), x) \leftarrow B(x)$$

$$c_6 : A(f(x)) \leftarrow B(x), C(x)$$

function-rule

Extending the calculus for $\mathcal{ELHI}$

- consider the clauses
  - $c_5 : R(f(x), x) \leftarrow B(x), c_7 : D(f(x)) \leftarrow B(x)$ (related to $B \sqsubseteq \exists R^-.D$)
  - $c_6 : A(f(x)) \leftarrow B(x), C(x)$ (obtained by the function rule)

Extending the calculus for $\mathcal{ELHI}$

- consider the clauses
  - $c_5 : R(f(x), x) \leftarrow B(x), c_7 : D(f(x)) \leftarrow B(x)$ (related to $B \sqsubseteq \exists R^{-}.D$)
  - $c_6 : A(f(x)) \leftarrow B(x), C(x)$ (obtained by the function rule)
- the same function f is associated with more than two clauses
  - $\Rightarrow$ we extend the shrinking rule to the *n-shrinking* which involves n side premises

Extending the calculus for $\mathcal{ELHI}$

- consider the clauses
  - $c_5 : R(f(x), x) \leftarrow B(x), c_7 : D(f(x)) \leftarrow B(x)$ (related to $B \sqsubseteq \exists R^-.D$)
  - $c_6 : A(f(x)) \leftarrow B(x), C(x)$ (obtained by the function rule)
- the same function f is associated with more than two clauses
  - $\Rightarrow$ we extend the shrinking rule to the *n-shrinking* which involves n side premises

$q : Q(x) \leftarrow R(y, x), A(y), D(y)$

n-shrinking

$c_2 : R(f(x), x) \leftarrow B(x), \quad c_6 : A(f(x)) \leftarrow B(x), C(x), \quad c_7 : D(f(x)) \leftarrow B(x)$

$q' : Q(x) \leftarrow B(x), C(x)$

Rapid for $\mathcal{ELHI}$

given the input $\mathcal{EL}$- clauses of the form $A(x) \leftarrow R(x,y) \wedge C(y)$ we apply exhaustively

- n-shrinking and unfolding to obtain clauses of the form $A(x) \leftarrow \bigwedge_i B_i(x)$
- function rule to obtain clauses $A(f(x)) \leftarrow \bigwedge_i B_i(x)$

Rapid for $\mathcal{ELHI}$

given the input $\mathcal{EL}$- clauses of the form $A(x) \leftarrow R(x, y) \wedge C(y)$ we apply exhaustively

- n-shrinking and unfolding to obtain clauses of the form $A(x) \leftarrow \bigwedge_i B_i(x)$
- function rule to obtain clauses $A(f(x)) \leftarrow \bigwedge_i B_i(x)$

- extend the TBox with the newly derived clauses
- apply calculus for DL-Lite on the input query and the extended TBox

Evaluation

- experimental evaluation over large scale ontologies
- we used $\mathcal{ELHI}$ versions of
  - NASA SWEET, S (http://sweet.jpl.nasa.gov/ontology/)
  - periodic table, C (http://www.cs.man.ac.uk/ stevensr/ontology/)
  - OBO protein, P

Table: Statistics of the used test ontologies

$\mathcal{O}$	concepts	roles	GCIs	RIAs
S	4298	519	6004	372
C	4282	22	9564	15
P	37560	6	52383	0

- we compared Rapid against Requiem, Clipper

Evaluation

	Time (ss:mm.msec)			Rewriting size		
$\mathcal{O}$	Rapid	Requiem	Clipper	Rapid	Requiem	Clipper
S	.08	.17	13.67	172	298	171
	.16	.36	13.32	473	1523	367
	.02	.44	13.79	629	1674	518
	.05	1.09	13.32	1065	2861	949
	.04	38.08	13.30	1075	18716	959
C	.06	3:36.54	21.42	1103	6800	2892
	.05	3:40.24	19.73	879	6941	2892
	.09	3:47.85	18.95	1653	6889	2892
	.08	3:31.29	20.48	1609	8077	2849
	.15	9:10.73	20.33	1743	57054	2893
P	4.80	t/o	1:42.08	51641	t/o	51641
	29.73	t/o	1:42.30	52877	t/o	52877
	2.59	t/o	1:45.35	51614	t/o	51614
	15.71	t/o	1:43.19	52407	t/o	52407
	13:51.97	t/o	1:47.10	79427	t/o	105950

Conclusions

- extended the calulus of Rapid for DL-Lite$_{\rightsquigarrow}$ an efficient resolution-based rewriting algorithm for $\mathcal{ELHI}$
- optimised resolution strategy, avoids well-known inefficiencies of resolution
- experimental evaluation shows that Rapid requires msecs for large scale ontologies

.

Future work

- extend our calculus for non-Horn DLs
- testing our system for query answering

Future work

- extend our calculus for non-Horn DLs
- testing our system for query answering
  - we have intergrated our rewriting in the tool presented in (Stoilos et al. Repairing ontologies for incomplete reasoners, ISWC 2011) for repairing ontologies
  - used OWLim to perform query answering (for real world ontologies, NCI, Galen)