

Conjunctive Queries for \mathcal{EL} with Role Composition

Markus Krötzsch

Sebastian Rudolph

Institute AIFB, Universität Karlsruhe (TH)

DL Workshop 2007

Presentation for the *DL 2007 paper of the same title* ([link](#))

Published under the terms of the Creative Common Attribution License 3.0 (CC-BY-3.0).

- OWL 1.1 envisions complex role inclusion axioms (RIAs)
“*hasParent* \circ *hasBrother* \sqsubseteq *hasUncle*”
- Conjunctive queries as major data access paradigm for DLs
- Tractable DLs required for data-intensive applications

→ Can we support conjunctive queries for $\mathcal{EL}^{++(-)}$?

- OWL 1.1 envisions complex role inclusion axioms (RIAs)
“hasParent o hasBrother \sqsubseteq hasUncle”
- Conjunctive queries as major data access paradigm for DLs
- Tractable DLs required for data-intensive applications

→ Can we support conjunctive queries for $\mathcal{EL}^{++(-)}$?

Table 1. Complexity of Tractable Fragments

Language	Reasoning Problems	Taxonomic Complexity	Data Complexity	Query Complexity	Combined Complexity
DL-Lite	Ontology Consistency, Concept Satisfiability, Concept Subsumption, Instance Checking,	In PTIME	In LOGSPACE	Not Applicable	In PTIME
	Conjunctive Query Answering	In PTIME	In LOGSPACE	NP-complete	NP-complete
EL++	Ontology Consistency, Concept Satisfiability, Concept Subsumption, Instance Checking	PTIME-complete	PTIME-complete	Not Applicable	PTIME-complete
	Conjunctive Query Answering	Open	PTIME-hard	Open	Open
	Ontology Consistency, Concept Satisfiability,	In EXPTIME	PTIME-complete	Not Applicable	In EXPTIME

Hardness of Querying

Conjunctive querying is undecidable for \mathcal{EL}^{++}

Cyclic dependencies in RBox are problematic \leadsto restrict RBoxes.



Hardness of Querying

Conjunctive querying is undecidable for \mathcal{EL}^{++}

Cyclic dependencies in RBox are problematic \leadsto restrict RBoxes.

With acyclic RBoxes we obtain:

- Query complexity: NP-hard when there is an ABox (*known*)



Hardness of Querying

Conjunctive querying is undecidable for \mathcal{EL}^{++}

Cyclic dependencies in RBox are problematic \leadsto restrict RBoxes.

With acyclic RBoxes we obtain:

- Query complexity: NP-hard when there is an ABox (*known*)
- Schema/data complexity: P-hard since \mathcal{EL} is (*known*)



Hardness of Querying

Conjunctive querying is undecidable for \mathcal{EL}^{++}

Cyclic dependencies in RBox are problematic \leadsto restrict RBoxes.

With acyclic RBoxes we obtain:

- Query complexity: NP-hard when there is an ABox (*known*)
- Schema/data complexity: P-hard since \mathcal{EL} is (*known*)
- Comb. complexity: PSPACE-hard when there is \exists and RBox (*new*)



Goal

Establish according
according upper bounds.

Property 1

\mathcal{EL} knowledge bases admit **canonical models** (Horn logic).

\leadsto no case distinctions in query answering

\leadsto query match with canonical model establishes query entailment

Property 1

\mathcal{EL} knowledge bases admit **canonical models** (Horn logic).

~> no case distinctions in query answering

~> query match with canonical model establishes query entailment

Property 2

Elements of canonical models described by **single concept names**.

~> all combinations of properties explicit in KB axioms

Property 1

\mathcal{EL} knowledge bases admit **canonical models** (Horn logic).

↪ no case distinctions in query answering

↪ query match with canonical model establishes query entailment

Property 2

Elements of canonical models described by **single concept names**.

↪ all combinations of properties explicit in KB axioms

Property 3

Elements in the model are **uniquely generated** by zero or more \exists -implications.

Cross-relations between elements only via constants.

How can a match of a query be represented?

Elements of the canonical model sufficiently described by:

- their “main” class and
- their generating path.

How can a match of a query be represented?

Elements of the canonical model sufficiently described by:

- their “main” class and
- their generating path.

Proof graph

- Nodes: elements of query (variables/constants)
- Node labels: single atomic concepts (“main” class)
- Edges: paths between elements (possible role chains)

Algorithm Overview

Step 1: Factorise query.

Algorithm Overview

Step 1: Factorise query.

Step 2: Guess node labels and generation path of variable nodes.

Automata for TBox subsumptions

Yielding all possible role paths between elements of concepts C and D .

Algorithm Overview

Step 1: Factorise query.

Step 2: Guess node labels and generation path of variable nodes.

Automata for TBox subsumptions

Yielding all possible role paths between elements of concepts C and D .

Step 3: Verify entailment of node labels.

Algorithm Overview

Step 1: Factorise query.

Step 2: Guess node labels and generation path of variable nodes.

Automata for TBox subsumptions

Yielding all possible role paths between elements of concepts C and D .

Step 3: Verify entailment of node labels.

Step 4: Guess decomposition of complex roles.

Query $R(x, y), S(y, z), T(x, z)$

$T_1 \circ T_2 \circ T_3 \sqsubseteq T \rightsquigarrow$ how to distribute T along path $x \rightarrow y \rightarrow z$?

Algorithm Overview

Step 1: Factorise query.

Step 2: Guess node labels and generation path of variable nodes.

Automata for TBox subsumptions

Yielding all possible role paths between elements of concepts C and D .

Step 3: Verify entailment of node labels.

Step 4: Guess decomposition of complex roles.

Query $R(x, y), S(y, z), T(x, z)$

$T_1 \circ T_2 \circ T_3 \sqsubseteq T \rightsquigarrow$ how to distribute T along path $x \rightarrow y \rightarrow z$?

Step 5: Verify entailment of role atoms.

Automata for role entailment

Yielding all possible role paths entailing role R .

\rightsquigarrow Compute role entailment by intersection of automata languages.

Complexity results

	Variable parts:				Complexity
	Query	R	T	A	
Combined complexity	●	●	●	●	PSPACE-complete
Query complexity	●				NP-complete
Schema complexity		●	●	●	PTIME-complete
Data complexity				●	PTIME-complete

What's next?

- Devise implementation strategies/goal-directed approaches.
- Extension to other Horn-DLs.
- Introduction of further OWL1.1 features.