Temporal Conjunctive Query Answering via Rewriting

Lukas Westhofen[1], Jean Christoph Jung[2], Daniel Neider[2,3]

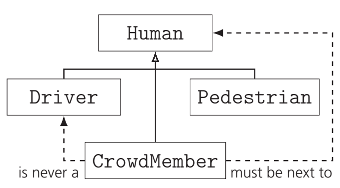
[1]German Aerospace Center (DLR e.V.), Institute of Systems Engineering for Future Mobility, Oldenburg, Germany
[2]TU Dortmund University, Dortmund, Germany
[3]Research Center Trustworthy Data Science and Security, University Alliance Ruhr, Dortmund, Germany
lukas.westhofen@dlr.de, jean.jung@tu-dortmund.de, daniel.neider@tu-dortmund.de

Setting

Ontology ($\mathcal{O}$)

Expressive Description Logics (DLs) model complex operational domain:

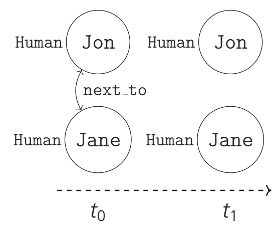
- $\texttt{Human} \equiv \texttt{Driver} \sqcup \texttt{Pedestrian}$ // any human must either be a driver or a pedestrian
- $\texttt{CrowdMember} \equiv \texttt{Human} \sqcap \exists\texttt{next\_to.Human}$ // crowd members are humans next to others
- $\texttt{CrowdMember} \sqsubseteq \neg\texttt{Driver}$ // crowd members are never drivers



Data ($\mathcal{D}$)

Systems (e.g., automated driving systems) gather temporal data in operation:

- t_0: $\texttt{Human(Jon)}, \texttt{Human(Jane)}, \texttt{next\_to(Jon, Jane)}, \texttt{next\_to(Jane, Jon)}$
- t_1: $\texttt{Human(Jon)}, \texttt{Human(Jane)}$



Temporal Conjunctive Query (TCQ) and Certain Answers

TCQ Φ: $\Box\texttt{Human}(x) \wedge \Diamond\texttt{Pedestrian}(x)$ // everything always being a human and eventually a pedestrian

Search in the Temporal Knowledge Base (TKB) $\mathcal{K} = (\mathcal{O}, \mathcal{D})$: $\text{cert}_{\mathcal{K}}(\Phi) = \{\texttt{Jane}, \texttt{Jon}\}$

. . . More Formally

TKBs: $\mathcal{K} = (\mathcal{O}, (\mathcal{D}_i)_{i\in\{0,\dots,n\}})$

$\mathcal{O}$ is a set of DL, e.g., $\mathcal{ALC}$, concepts:
$C ::= \texttt{A} \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall\texttt{r}.C \mid \exists\texttt{r}.C$
$\mathcal{D}_i$ is a set of assertions:
$\texttt{A(i)}, \texttt{r(i}_1, \texttt{i}_2\texttt{)}$ over individuals $\texttt{i}, \texttt{i}_1, \texttt{i}_2$

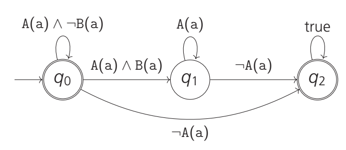
TCQs: $\Phi ::= \neg\Phi \mid \Phi \wedge \Phi \mid \bigcirc\Phi \mid \Phi\,\mathcal{U}\,\Phi \mid \varphi$

φ is a conjunction of atoms $\texttt{A}(v)$, $\texttt{r}(v_1, v_2)$ where v, v_1, v_2 are

- individuals $\texttt{i}$,
- answer variables $\vec{x}$, or
- existentially quantified variables $\vec{y}$.

cert$_{\mathcal{K}}(\Phi)$: Assignments to $\vec{x}$ guaranteeing Φ under any interpretation of $\mathcal{K}$ (written '$\mathcal{K} \models \Phi$').

State of the Art using Finite Automata

Entailment of TCQ	⇝	Unsatisfiability of negated TCQ	⇝	Finite Automata	⇝	Reachability Check
$\mathcal{K} \models \Box\texttt{A(a)} \wedge \Diamond\texttt{B(a)}$		$\mathcal{K}, \neg(\Box\texttt{A(a)} \wedge \Diamond\texttt{B(a)})$ unsatisfiable				Return true iff *no* final state is reachable on data.

A Novel Approach using Rewriting

Idea: Reduce unnecessary dependency on UCQs by rewriting Φ to equivalent Φ'.

Assume Negation Normal Form

- Rule 1: $\bigcirc\Phi_1 \vee \bigcirc\Phi_2 \rightsquigarrow \bigcirc(\Phi_1 \vee \Phi_2)$
- Rule 2: $\Phi_1 \vee (\Phi_2 \wedge \Phi_3) \rightsquigarrow (\Phi_1 \vee \Phi_2) \wedge (\Phi_1 \vee \Phi_3)$
- Rule 3: $\Phi_1\,\mathcal{U}\,\Phi_2 \rightsquigarrow \Phi_2 \vee (\Phi_1 \wedge \bigcirc(\Phi_1\,\mathcal{U}\,\Phi_2))$
- Rule 4: $\Phi_1\,\mathcal{R}\,\Phi_2 \rightsquigarrow \Phi_2 \wedge (\Phi_1 \vee \bullet(\Phi_1\,\mathcal{R}\,\Phi_2))$
- Rule 5: $\bullet\Phi \rightsquigarrow \text{last} \vee \bigcirc\Phi$

$$\Phi' = \bigwedge_{k\in\{1,\dots,u\}} \Phi_k \wedge \quad // \ \Phi_k \text{ non-temporal}$$
$$\bigwedge_{l\in\{u+1,\dots,v\}} \Phi_l \wedge \quad // \ \Phi_l = \bigcirc\Phi'_l$$
$$\bigwedge_{m\in\{v+1,\dots,w\}} \Phi_m, \quad // \ \Phi_m = \bigcirc\Phi^1_m \vee \Phi^2_m, \ \Phi^2_m \text{ non-temporal}$$

This normal form allows to safely combine disjunctions:

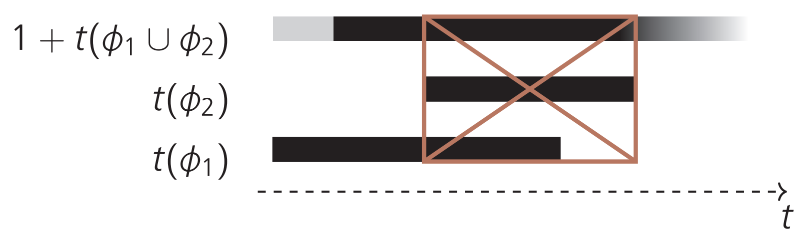
$$\text{cert}_{\mathcal{K}_{\geq j}}(\Phi') = \bigcap_{k\in\{1,\dots,u\}} \text{cert}_{K_j}(\Phi_k) \cap \bigcap_{l\in\{u+1,\dots,v\}} \text{cert}_{K_{\geq j+1}}(\Phi'_l) \cap \bigcap_{m\in\{v+1,\dots,w\}} \text{cert}_{K_{\geq j+1}}(\Phi^1_m) \cup \text{cert}_{K_j}(\Phi^2_m)$$

An Efficient Fragment

Fragment $\text{TCQ}_{\text{CQ}^*} \subset \text{TCQ}$ guarantees answering to only require CQ answering (no UCQs).

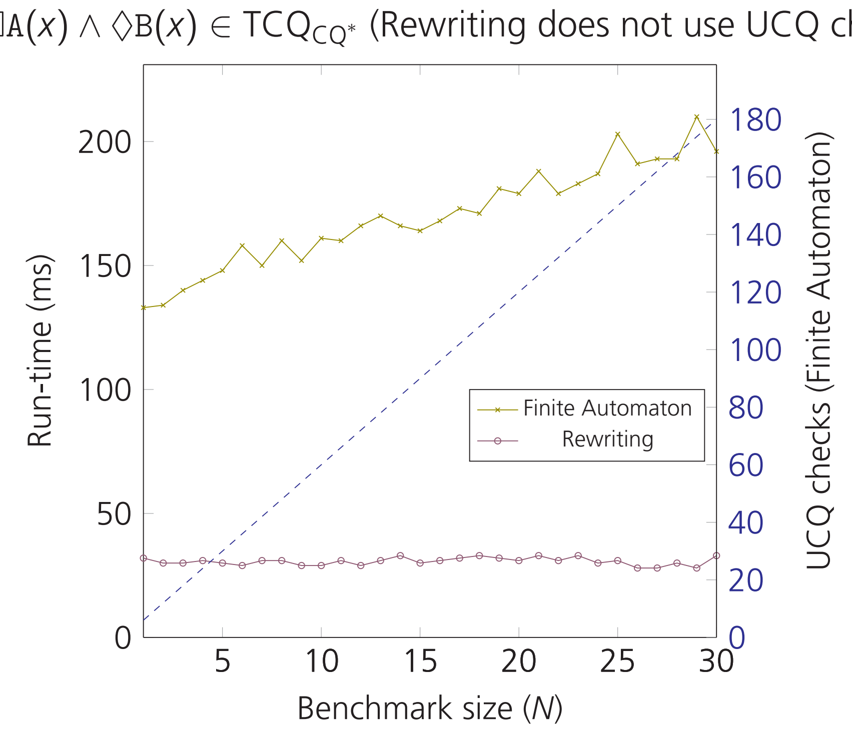
Not obvious:
$(\{\top \sqsubseteq \texttt{A}\sqcup\texttt{B}\}, (\{\texttt{A(a)}\}, \emptyset, \{\texttt{B(a)}\})) \models \Diamond(\texttt{A(a)} \wedge \bigcirc\texttt{B(a)})$
does require UCQs!

Idea: For $\Phi_1 \vee \Phi_2$ and $\Phi_1\,\mathcal{U}\,\Phi_2$, disallow temporally overlapping disjunctions:

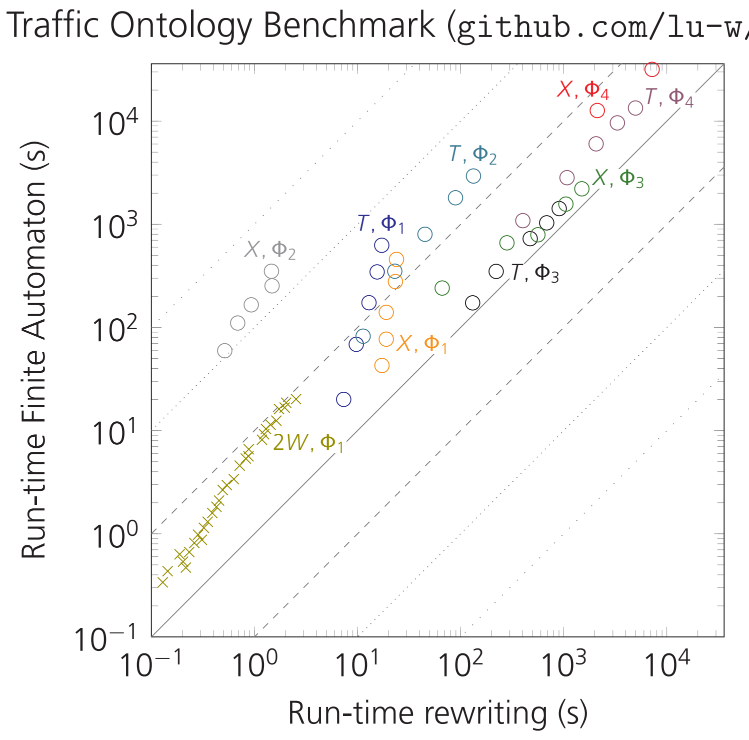


Evaluation

On $\Box\texttt{A}(x) \wedge \Diamond\texttt{B}(x) \in \text{TCQ}_{\text{CQ}^*}$ (Rewriting does not use UCQ checks):



On the Traffic Ontology Benchmark (github.com/lu-w/tobm):



Main Findings

Order of magnitude speed-up compared to finite automata	Why? Reduction of (expensive) UCQ answering	Why? Automata translation does not push introduced negation inwards

Bonus: Rewriting approach enables defining a fragment of TCQ that guides users to efficiently answerable queries

More Information:

Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center