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

Abstract

Querying temporal data has recently gained traction in several artificial intelligence applications. As operational domains of intelligent agents are constantly being expanded, there is a strong need for representing domain knowledge. This comes in the form of ontologies, which are predominantly expressed in description logics and enrich time-stamped data to temporal knowledge bases. For modeling highly complex system environments, expressive description logics are often the formalism of choice. Querying such temporal knowledge bases is a challenging task, but recently a first practical solution has been put forward. We propose a novel approach to the query answering problem based on two well-known rewriting rules from temporal logic. After a careful theoretical analysis of our algorithm, we show in a practical evaluation on several benchmarks that it outperforms state of the art, sometimes by orders of magnitude. Based on our findings, we also propose a fragment of temporal conjunctive queries which guides users towards well-performing queries.

1 Introduction

Today's AI-based systems produce large amounts of data during operation in the field. Take, for example, automated driving companies like Waymo, who had an accumulated total of 7.14 million miles of driving data on December 2023 (Kusano et al. 2023). These data are typically temporal (systems operate over time) and the underlying data model is complex (tasks are performed in highly diverse, relational environments). For continuous safety assurance, manufacturers require to thoroughly analyze a non-negligible fraction of the produced data, such as critical near-accidents.

A promising solution to logically analyzing these data is offered by *temporal conjunctive queries (TCQs)* over *temporal knowledge bases (TKBs)* (Artale et al. 2013; Calvanese et al. 2023). TKBs augment temporal, relational data with background knowledge in form of logical axioms. They consist of two elements: an ontology $\mathcal{O}$ typically formulated in some *description logic (DL)* and a finite sequence of relational data $(\mathcal{D}_i)_{i\in\{0,\dots,n\}}$. Depending on the application, the DL ontology represents background knowledge of varying complexity, where the intricate contexts of intelligent systems often warrant high expressiveness. It can, for example, contain the facts that any human must either be a driver or a pedestrian ($\texttt{Human} \equiv \texttt{Driver} \sqcup \texttt{Pedestrian}$), crowd members are humans next to other humans ($\texttt{CrowdMember} \equiv \texttt{Human} \sqcap \exists \texttt{next\_to}.\texttt{Human}$), and they are never drivers ($\texttt{CrowdMember} \sqsubseteq \neg\texttt{Driver}$). The data component assigns classes and relationships to individuals – perceived objects in the world. The temporal data can assert, at some time i, the humans Jon and Jane to be next to each other: $\texttt{Human}(\texttt{Jon}), \texttt{Human}(\texttt{Jane}), \texttt{next\_to}(\texttt{Jon}, \texttt{Jane}), \texttt{next\_to}(\texttt{Jane}, \texttt{Jon}) \in \mathcal{D}_i$.

We examine querying of temporal data under expressive DL ontologies with *temporal conjunctive queries (TCQs)*, a popular combination of *linear temporal logic over finite traces (LTL$_f$)* and *conjunctive queries (CQs)*, a standard database query language. For example, $\Diamond\texttt{Pedestrian}(x)$ asks for anything eventually being a pedestrian. Answers include all crowd members (Jane and Jon) without the data explicitly asserting them to be pedestrians.

The described setup has been thoroughly studied from a theoretical perspective, and based on that recently a first prototype implementation was proposed. For the former, Baader et al. provide a complexity analysis for the more general case of infinite traces with past-time operators showing that answering temporal queries over the expressive DL $\mathcal{ALC}$ is ExpTime-complete (Baader, Borgwardt, and Lippmann 2013). In a nutshell, the provided decision procedure is a Turing reduction to standard (that is, non-temporal) query answering under ontologies. The idea is to compile the temporal aspects of the query into a *finite automaton (FA)* whose transitions are checked for satisfiability by answering *unions of conjunctive queries (UCQs)*. Recently, Westhofen et al. implemented this approach for the case of finite traces and TCQs with metric temporal operators (2024a; 2024b). This resulted in the tool Topllet which exhibited promising initial results. Interestingly, it was already recognized that the lack of good UCQ answering engines (in the non-temporal setting) can lead to performance issues; indeed, a main contribution in (Westhofen et al. 2024a) was a pre-processing step using optimized CQ engines to avoid answering UCQs.

Our main contribution is a novel approach to answering TCQs over temporal knowledge bases with ontologies formulated in expressive DLs based on *rewritings*, which is presented in Section 4. The idea is to rewrite TCQs into a certain normal form exploiting the well-known expansion

law for the temporal 'until'-operator $\mathcal{U}$ as well as the fact that the temporal 'next'-operator $\bigcirc$ distributes over disjunctions. This normal form allows us to inductively combine answers to UCQs at the current time point with answers for the remaining time points. From a theoretical perspective, we show that our approach is worst-case optimal, so it is ExpTime-complete over ontologies formulated in $\mathcal{ALC}$.

While our approach is a Turing reduction to standard UCQ answering under ontologies as well, we demonstrate that it does not ask the same UCQs as the FA-based approach and that, in fact, there are queries for which the latter approach relies on UCQ answering and we do not. We implemented our rewriting approach and evaluated it on large-scale benchmarks for expressive DLs. A comparison to the tool of Westhofen et al. in Section 5 shows improvements on all benchmarks, in some cases by two orders of magnitude. Our evaluation supports the hypothesis that better run times are mainly explained by a more precise use of UCQs.

Motivated by these empirical observations, Section 6 formalizes, as our second contribution, a non-trivial fragment of TCQs, called TCQ_{CQ^*}, for which our approach *never* resorts to the UCQ engine. We envision that this fragment can help to guide users towards well-performing TCQs, even when they demand expressive DLs.

2 Related Work

Directly related work on temporally querying *expressive* DLs other than the discussed (Baader, Borgwardt, and Lippmann 2013; Westhofen et al. 2024a,b) is sparse. We acknowledge, however, a variety of advances on generally integrating temporal logics into DLs, see Artale and Franconi (2000), Lutz, Wolter, and Zakharyaschev (2008), and Artale et al. (2014; 2017) for excellent surveys.

A second line of work examines the case of restricted expressiveness in the DL, e.g., DL-Lite (Borgwardt, Lippmann, and Thost 2013) and $\mathcal{EL}$ (Borgwardt and Thost 2015; Gutiérrez-Basulto, Jung, and Kontchakov 2016). For several of these lightweight DLs it was also studied how to rewrite temporal queries and ontology into standard query languages (without ontologies) (Borgwardt, Lippmann, and Thost 2015; Artale et al. 2022, 2013). The field has advanced significantly and efficient tools are being implemented, e.g., by rewriting to temporal Datalog (Wang et al. 2022) or SQL (Thost, Holste, and Özçep 2015; Kalayci et al. 2018).

One of the core parts of our approach is CQ answering, for which practical applicability has been achieved by heuristics, e.g., as described by Sirin (2006) and implemented in Pellet (Sirin et al. 2007). Besides Pellet, a wide range of software tools offers CQ answering in expressive DLs (Parsia et al. 2017), yet none considers temporal queries.

3 Preliminaries

d As motivated in the introduction, a TKB $\mathcal{K}$ connects temporal data $(\mathcal{D}_i)_{i\in\{0,\dots,n\}}$ with background knowledge $\mathcal{O}$ and is thus a tuple $\mathcal{K} = (\mathcal{O}, (\mathcal{D}_i)_{i\in\{0,\dots,n\}})$. For simplicity, we also use $\mathcal{K}_i$ to refer to the non-temporal *knowledge base (KB)* $(\mathcal{O}, \mathcal{D}_i)$. The first component, $\mathcal{O}$, generally enables to store concept inclusions $C \sqsubseteq D$ (or $C \equiv D$ for $C \sqsubseteq D$ and $D \sqsubseteq C$), where the concrete syntax of C and D depends on the employed Description Logic. First, we require sets $\mathsf{N_C}, \mathsf{N_R}, \mathsf{N_I}$ of concepts, roles, and individual names, whose elements are subsequently denoted in typewriter font. For conciseness, we focus on the expressive DL $\mathcal{ALC}$, where $C ::= \mathtt{A} \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall \mathtt{r}.C \mid \exists \mathtt{r}.C$ for $\mathtt{A} \in \mathsf{N_C}$ and $\mathtt{r} \in \mathsf{N_R}$ (Baader et al. 2007). We have already encountered $\mathcal{ALC}$ concept inclusions in Section 1, e.g., $\mathtt{CrowdMember} \sqsubseteq \neg\mathtt{Driver}$ with $\mathtt{CrowdMember}, \mathtt{Driver} \in \mathsf{N_C}$. The second component, $(\mathcal{D}_i)_{i\in\{0,\dots,n\}}$, represents a temporal sequence, where each element $\mathcal{D}_i$ stores data as assertions from (pairs of) individuals to concepts and roles, i.e., is of the form $\mathtt{r}(\mathtt{a},\mathtt{b})$ and $\mathtt{C}(\mathtt{a})$ for $\mathtt{a},\mathtt{b} \in \mathsf{N_I}$, $\mathtt{r} \in \mathsf{N_R}$, and $\mathtt{A} \in \mathsf{N_C}$. The introduction used $\mathtt{Human}(\mathtt{Jon})$ as an example, where $\mathtt{Human} \in \mathsf{N_C}$ and $\mathtt{Jon} \in \mathsf{N_I}$. Overall, the ontology and data of Section 1 can be used in a TKB over two time points as $\mathcal{K}_{ex} = (\mathcal{O}, \mathcal{D})$ with $\mathcal{O} = (\{\mathtt{Human} \equiv \mathtt{Driver} \sqcup \mathtt{Pedestrian}, \mathtt{CrowdMember} \equiv \mathtt{Human} \sqcap \exists \mathtt{next\_to}.\mathtt{Human}, \mathtt{CrowdMember} \sqsubseteq \neg\mathtt{Driver}\}$, $\mathcal{D} = (\{\mathtt{Human}(\mathtt{Jon}), \mathtt{Human}(\mathtt{Jane}), \mathtt{next\_to}(\mathtt{Jon}, \mathtt{Jane}), \mathtt{next\_to}(\mathtt{Jane}, \mathtt{Jon})\}, \{\mathtt{Human}(\mathtt{Jon}), \mathtt{Human}(\mathtt{Jane})\})$.

For defining semantics of TKBs, we rely on the usual way of (temporal) interpretations.

Definition 1 ((Temporal) Interpretation)**.** *An* interpretation $\mathcal{I}$ *is a tuple* $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$*, where* $\Delta^{\mathcal{I}}$ *is a domain and* $\cdot^{\mathcal{I}}$ *is a function assigning a set* $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ *to every* $A \in \mathsf{N_C}$*, a binary relation* $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ *to every role name* $r \in \mathsf{N_R}$*, and an element* $a^{\mathcal{I}}$ *to every* $a \in \mathsf{N_I}$*. A* temporal interpretation $\mathfrak{I}$ *is a finite sequence* $\mathfrak{I} = (\mathcal{I}_i)_{i\in\{0,\dots,m\}}$ *over a fixed domain* Δ *such that* $a^{\mathcal{I}_i} = a^{\mathcal{I}_j}$*, for all* $a \in \mathsf{N_I}$ *and* $0 \le i, j \le m$*.*

Since only the data component is temporal, we use standard interpretations $\mathcal{I}$ for the semantics of $\mathcal{ALC}$-concepts:

$$(\neg\mathtt{C})^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus \mathtt{C}^{\mathcal{I}}, (\mathtt{C} \sqcap \mathtt{D})^{\mathcal{I}} = \mathtt{C}^{\mathcal{I}} \cap \mathtt{D}^{\mathcal{I}}, (\mathtt{C} \sqcup \mathtt{D})^{\mathcal{I}} = \mathtt{C}^{\mathcal{I}} \cup \mathtt{D}^{\mathcal{I}}$$

$$(\forall \mathtt{r}.\mathtt{C})^{\mathcal{I}} = \{c \in \Delta^{\mathcal{I}} \mid \forall d \in \Delta^{\mathcal{I}}.\, (c,d) \in \mathtt{R}^{\mathcal{I}} \to d \in \mathtt{C}^{\mathcal{I}}\}$$

$$(\exists \mathtt{r}.\mathtt{C})^{\mathcal{I}} = \{c \in \Delta^{\mathcal{I}} \mid \exists d \in \Delta^{\mathcal{I}}.\, (c,d) \in \mathtt{R}^{\mathcal{I}} \wedge d \in \mathtt{C}^{\mathcal{I}}\}$$

For lifting this to concept inclusions, we write $\mathcal{I} \models C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a model of an ontology $\mathcal{O}$, written $\mathcal{I} \models \mathcal{O}$, if $\mathcal{I}$ models all concept inclusions in $\mathcal{O}$. For the data component, we define $\mathcal{I} \models \mathtt{A}(\mathtt{a})$ if $\mathtt{a}^{\mathcal{I}} \in \mathtt{A}^{\mathcal{I}}$, $\mathcal{I} \models \mathtt{r}(\mathtt{a},\mathtt{b})$ if $(\mathtt{a}^{\mathcal{I}}, \mathtt{b}^{\mathcal{I}}) \in \mathtt{r}^{\mathcal{I}}$, and $\mathcal{I} \models \mathcal{D}$ if all assertions in $\mathcal{D}$ adhere to the previous definition. Finally, a temporal interpretation $\mathfrak{I} = (\mathcal{I}_i)_{i\in\{0,\dots,m\}}$ is a model of a TKB $\mathcal{K} = (\mathcal{O}, (\mathcal{D}_i)_{i\in\{0,\dots,n\}})$, written $\mathfrak{I} \models \mathcal{K}$, if $m = n$ and $\mathcal{I}_i \models \mathcal{D}_i$ and $\mathcal{I}_i \models \mathcal{O}$ for all $i \in \{0,\dots,n\}$.

One of the core reasoning tasks over TKBs is query entailment. As motivated in Section 1, we build on Westhofen et al. (2024a), where linear temporal operators are over finite traces are combined with CQs, leading to TCQs like $\Phi_{ex} = \Diamond\mathtt{Pedestrian}(x)$. In contrast to Westhofen et al., we do not consider metric operators, as they are syntactic sugar (over discrete, finite traces, they can be simulated by exponentially many next formulae (Li, Vardi, and Rozier 2019)).

Definition 2 (Syntax of TCQs)**.** *For a countably infinite set of variable names* $\mathsf{N_V}$*, a CQ* φ *is of the form* $\varphi = \exists\vec{y}.\psi(\vec{x},\vec{y})$*, where* $\vec{x},\vec{y}$ *are tuples from* $\mathsf{N_V}$ *and* ψ *is a conjunction of atoms, i.e., concepts* $\mathtt{A}(t)$ *and* $\mathtt{r}(t,t')$ *with* $\mathtt{A} \in$

$\mathsf{N_C}$, $\mathtt{r} \in \mathsf{N_R}$, *and* $t, t' \in \vec{x} \cup \vec{y} \cup \mathsf{N_I}$. $\mathsf{Atoms}(\varphi)$ *denotes the set of atoms in* φ. *A TCQ* Φ *is a formula of the form*

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

where φ *ranges over CQs.*

Thus, TCQs augment $\mathrm{LTL_f}$ (De Giacomo and Vardi 2013) with CQs as atoms. $\mathsf{Ind}(\Phi)$ and $\mathsf{Var}(\Phi)$ are the sets of individuals and variables in Φ, respectively. Specifically, the tuples $\vec{x}$ and $\vec{y}$ contain elements from the sets of *answer variables* $\mathsf{AVar}(\Phi)$ and *quantified variables* $\mathsf{QVar}(\Phi)$ (which are not returned as answers), respectively. For our example Φ_{ex}, $\mathsf{AVar}(\Phi) = \{x\}$ and $\mathsf{QVar}(\Phi) = \emptyset$. We use additional operators as usual: Weak next ($\CIRCLE$) is the dual of strong next: $\CIRCLE\Phi \equiv \neg\bigcirc\neg\Phi$. Moreover, $\Phi_1 \vee \Phi_2 \equiv \neg(\neg\Phi_1 \wedge \neg\Phi_2)$, $\mathsf{false} \equiv \exists x.\mathtt{A}(x) \wedge \neg\exists x.\mathtt{A}(x)$ for an $\mathtt{A} \in \mathsf{N_C}$, $\mathsf{true} \equiv \neg\mathsf{false}$, $\Diamond\Phi \equiv \mathsf{true}\mathcal{U}\Phi$, $\Box\Phi \equiv \neg\Diamond\neg\Phi$, and $\Phi_1 \mathcal{R} \Phi_2 \equiv \neg(\neg\Phi_1 \mathcal{U} \neg\Phi_2)$. To avoid last or weak next as a temporal operator later on, we assume an auxiliary predicate last which is true only at the last time point (e.g., by the CQ $\exists x.\mathtt{Last}(x)$ and adding $\mathtt{Last}(a)$ to the last database). Finally, every TCQ can be brought in linear time into in *negation normal form (NNF)*, where negations appear only in front of CQs.

It remains to define TCQ semantics. We start with the semantics of its central part, CQs, for which we use homomorphisms from queries to domains of non-temporal interpretation. We first consider *Boolean* queries, i.e., without answer variables. For example, $\mathtt{Pedestrian}(\mathtt{Jon})$ is Boolean.

Definition 3 (Semantics of Boolean CQs)**.** *Given a Boolean CQ* φ *and an interpretation* $\mathcal{I}$*, we say that* $\mathcal{I}$ *is a model of* φ*, written as* $\mathcal{I} \models \varphi$*, if there is a function* $\pi\colon \mathsf{Var}(\varphi) \cup \mathsf{Ind}(\varphi) \to \Delta^{\mathcal{I}}$ *with 1.* $\pi(a) = a^{\mathcal{I}}$ *for all* $a \in \mathsf{Ind}(\varphi)$*, 2.* $\pi(t) \in \mathtt{C}^{\mathcal{I}}$ *for all* $\mathtt{C}(t)$ *in* φ*, and 3.* $(\pi(t), \pi(t')) \in \mathtt{r}^{\mathcal{I}}$ *for all* $\mathtt{r}(t, t')$ *in* φ.

Note that we do not consider infinite traces as examined by Baader, Borgwardt, and Lippmann (2013), which is motivated by our application of analyzing finite field data of intelligent systems. The semantics of TCQs is thus defined similar to $\mathrm{LTL_f}$ and relies on finite temporal interpretations.

Definition 4 (Semantics of Boolean TCQs)**.** *For a temporal interpretation* $\mathfrak{I} = (\mathcal{I}_i)_{i \in \{0,\dots,m\}}$ *and* $i \in \{0, \dots, m\}$*, we define the semantics of Boolean TCQs as:*

- $\mathfrak{I}, i \models \Phi$ *iff* $\mathcal{I}_i \models \Phi$*, if* Φ *is a Boolean CQ;*
- $\mathfrak{I}, i \models \neg\Phi$ *iff* $\mathfrak{I}, i \not\models \Phi$*;*
- $\mathfrak{I}, i \models \Phi_1 \wedge \Phi_2$ *iff* $\mathfrak{I}, i \models \Phi_1$ *and* $\mathfrak{I}, i \models \Phi_2$*;*
- $\mathfrak{I}, i \models \bigcirc\Phi$ *iff* $i < m$ *and* $\mathfrak{I}, i+1 \models \Phi$*;*
- $\mathfrak{I}, i \models \Phi_1 \mathcal{U} \Phi_2$ *iff there is a* $k \in [i, m]$ *such that* $\mathfrak{I}, k \models \Phi_2$ *and* $\mathfrak{I}, j \models \Phi_1$*, for all* $j \in [i, i+k)$.

The core problem of Boolean TCQs is entailment, which is the semantic basis of our work. Intuitively, it asks whether all models of the TKB satisfy the TCQ as well.

Definition 5 (Entailment of Boolean TCQs)**.** *For a TKB* $\mathcal{K}$ *and an TCQ* Φ*,* $\mathcal{K} \models \Phi$ *if for all temporal interpretations* $\mathfrak{I}$ *with* $\mathfrak{I} \models \mathcal{K}$ *it holds that* $\mathfrak{I}, 0 \models \Phi$.

For our example TKB $\mathcal{K}_{ex}$, we can assert that $\mathcal{K}_{ex} \models \Diamond\mathtt{Pedestrian}(\mathtt{Jon})$, as in any model of $\mathcal{K}_{ex}$ the individual $\mathtt{Jon}$ will be a $\mathtt{Pedestrian}$ at some time point.

In practice, users often require not only entailment checks but wish to retrieve guaranteed solutions, so-called certain answers, to non-Boolean queries that contain answer variables (similar to standard database querying). These can be conveniently defined via entailment.

Definition 6 (Certain Answers)**.** *Let* $\mathcal{K} = (\mathcal{O}, (\mathcal{D}_i)_{i \in \{0,\dots,n\}})$ *be a TKB,* Φ *an TCQ with answer variables* $\vec{x}$*, and* $\vec{a}$ *a tuple of individuals from* $\mathcal{K}$*, i.e.,* $\mathsf{Ind}(\mathcal{K}) \coloneqq \bigcup_{i=0,\dots,n} \mathsf{Ind}(\mathcal{D}_i)$*. We call* $\vec{a}$ *a* certain answer to Φ over $\mathcal{K}$ *if* $\mathcal{K} \models \Phi(\vec{a})$*, where the Boolean TCQ* $\Phi(\vec{a})$ *has the variables in* $\vec{x}$ *uniformly substituted by* $\vec{a}$.

The set of certain answers of a TCQ Φ over a TKB $\mathcal{K}$ is denoted by $\mathsf{cert}_{\mathcal{K}}(\Phi)$. Computing this set is the main reasoning task considered in this paper. For our example query Φ_{ex}, Section 1 showed that $\mathsf{cert}_{\mathcal{K}_{ex}}(\Phi_{ex}) = \{\mathtt{Jane}, \mathtt{Jon}\}$.

4 Rewriting-Based Answering of TCQs

We present a novel way to answer TCQs over TKBs, that is, solving the problem of computing the set $\mathsf{cert}_{\mathcal{K}}(\Phi)$, without construction of an automaton. We focus on the DL $\mathcal{ALC}$ for brevity, but the approach is applicable to any DL where UCQ entailment is decideable, e.g., $\mathcal{ALCI}$.

For motivation, it is instructive to consider $\Phi_\vee \coloneqq \bigcirc\mathtt{B}(x) \vee \bigcirc\mathtt{C}(x)$ over the TKBs $\mathcal{K}_\emptyset \coloneqq (\emptyset, (\mathcal{D}_i)_{i \in \{0,\dots,n\}})$ and $\mathcal{K}_\vee \coloneqq (\mathtt{A} \sqsubseteq \mathtt{B} \sqcup \mathtt{C}, (\emptyset, \mathtt{A}(\mathtt{a})))$. The certain answers over $\mathcal{K}_\emptyset$ can be computed in an inductive way by first computing certain answers to $\bigcirc\mathtt{B}(x)$ and $\bigcirc\mathtt{C}(x)$ separately and then taking their union. This is not the case for KBs with expressive ontologies. The certain answer of $\Phi_\vee$ over $\mathcal{K}_\vee$ is $\mathtt{a}$. Yet, if we check $\bigcirc\mathtt{B}(x)$ and $\bigcirc\mathtt{C}(x)$ separately, no answer is identified. The correct approach is to rewrite $\Phi_\vee(x)$ into $\bigcirc(\mathtt{B}(x) \vee \mathtt{C}(x))$, resolve $\bigcirc$, and answer $\mathtt{B}(x) \vee \mathtt{C}(x)$ jointly.

Our approach generalizes this idea by rewriting the query into a normal form that relies on two sets of rules: The standard expansion laws (Baier and Katoen 2008) and rules to correctly distribute $\bigcirc$ over $\vee$. They ensure that we can safely use inductive combination for temporal disjunctions and have *unions of possibly negated conjunctive queries (UnCQs)* in the non-temporal case, for which we show next how to give a Turing reduction to UCQ answering.

Answering Unions of Possibly Negated CQs

As our queries allow negation and disjunction, we rely on answering UnCQs as the base case. The algorithm $\mathsf{AnswerUnCQ}(\mathcal{K}, \Psi)$ hence computes the certain answers to a UnCQ $\Psi(\vec{x}) = \bigvee_{i=1,\dots,k} \varphi_i \vee \bigvee_{j=k+1,\dots,l} \neg\varphi_j$ over a non-temporal KB $\mathcal{K}$. It is defined via the established way of reduction to UCQ entailment (Baader, Borgwardt, and Lippmann 2013). Formally, $\mathsf{AnswerUnCQ}(\mathcal{K}, \Psi) \coloneqq \{\vec{c} \in \mathsf{Ind}(\mathcal{K})^{|\vec{x}|} \mid (\mathcal{O}, \mathcal{D} \cup \bigcup_{j=k+1,\dots,l} \mathsf{Atoms}(\varphi_j(\vec{c}))) \models \bigvee_{i=1,\dots,k} \varphi_i(\vec{c})\}$ with quantified variables uniquely named.

Lemma 1. *For a KB* $\mathcal{K}$ *and a union of possibly negated CQs* $\Psi = \bigvee_{i=1,\dots,k} \varphi_i \vee \bigvee_{j=k+1,\dots,l} \neg\varphi_j$ *it holds that* $\mathsf{AnswerUnCQ}(\Psi, \mathcal{K}) = \mathsf{cert}_{\mathcal{K}}(\Psi)$.

Note that $\mathsf{AnswerUnCQ}(\Psi, \mathcal{K})$ is a proven procedure from the literature for checking satisfiability of conjunctions of

1. $\bigcirc\Phi_1 \vee \bigcirc\Phi_2 \rightsquigarrow \bigcirc(\Phi_1 \vee \Phi_2)$
2. $\Phi_1 \vee (\Phi_2 \wedge \Phi_3) \rightsquigarrow (\Phi_1 \vee \Phi_2) \wedge (\Phi_1 \vee \Phi_3)$
3. $\Phi_1 \,\mathcal{U}\, \Phi_2 \rightsquigarrow \Phi_2 \vee (\Phi_1 \wedge \bigcirc(\Phi_1 \,\mathcal{U}\, \Phi_2))$
4. $\Phi_1 \,\mathcal{R}\, \Phi_2 \rightsquigarrow \Phi_2 \wedge (\Phi_1 \vee \CIRCLE(\Phi_1 \,\mathcal{R}\, \Phi_2))$
5. $\CIRCLE\Phi \rightsquigarrow \mathsf{last} \vee \bigcirc\Phi$

Figure 1: Transformation rules used by $\mathsf{Transform}(\Phi)$.

possibly negated CQs (Baader, Borgwardt, and Lippmann 2013, Proof of upper bound of Theorem 3.2) by leveraging the fact that $\mathcal{K} \models \Psi$ iff $\neg\Psi = \bigwedge_{i=1,\dots,k} \neg\varphi_i \wedge \bigwedge_{j=k+1,\dots,l} \varphi_j$ is unsatisfiable w.r.t. $\mathcal{K}$. Correctness of Lemma 1 is hence asserted.

Rewriting

We have previously demonstrated that temporal disjunctions are the main issue when naïvely inductively combining answers to a given TCQ. We approach this issue by rewriting the TCQ into a normal form first. This normal form ensures that for any disjunction we can just inductively combine answers to its disjuncts (if it is temporal) or apply the previously introduced procedure of AnswerUnCQ (if it is non-temporal). For this, we establish the following shape of the TCQ: Any temporal operator is wrapped into $\bigcirc$ formulae and any disjunction has at most one temporal disjunct.

For handling negations on CQ level directly, we assume the input Φ to Transform to be in NNF. Due to this, our transformation rules must, besides the primitives $\wedge, \bigcirc, \mathcal{U}$, also consider $\vee, \CIRCLE, \mathcal{R}$. To achieve our normal form, $\mathsf{Transform}(\Phi)$ applies the rules of Figure 1 to any subformula in Φ not occurring within a temporal operator ($\bigcirc$, $\CIRCLE$, $\mathcal{U}$, or $\mathcal{R}$), until no more rule can be fired. Rule 1 distributes next over disjunctions (as motivated earlier), Rule 2 uses distributivity of $\vee$ over $\wedge$ to establish a *conjunctive normal form (CNF)* over non-temporal and next formulae, and Rules 3 and 4 apply the expansion law. Rule 5 allows to disregard $\CIRCLE$ using last. In general and for the transformation rules specifically, we assume commutativity and associativity of $\vee$, i.e., $\Phi_1 \vee \Phi_2 = \Phi_2 \vee \Phi_1$ and $(\Phi_1 \vee \Phi_2) \vee \Phi_3 = \Phi_1 \vee (\Phi_2 \vee \Phi_3)$. For example, Rule 1 on $\bigcirc A(x) \vee B(x) \vee \bigcirc C(x)$ results in $B(x) \vee \bigcirc(A(x) \vee C(x))$.

$\mathsf{Transform}(\Phi)$ has two central properties (Lemma 2 and 3): It preserves original semantics and establishes a correct way of inductively computing $\mathsf{cert}_{\mathcal{K}_{\geq j}}(\Phi)$.

Lemma 2. *$\mathfrak{I}, 0 \models \Phi$ iff $\mathfrak{I}, 0 \models \mathsf{Transform}(\Phi)$ for any TCQ Φ and temporal interpretation $\mathfrak{I}$.*

Lemma 3. *Let Φ be a TCQ and $\Phi' := \mathsf{Transform}(\Phi)$. Then Φ' takes the form of a conjunction*

$$\textstyle\bigwedge_{k\in\{1,\dots,u\}} \Phi_k \wedge \bigwedge_{l\in\{u+1,\dots,v\}} \Phi_l \wedge \bigwedge_{m\in\{v+1,\dots,w\}} \Phi_m,$$

where each Φ_k is a UnCQ, each $\Phi_l = \bigcirc\Phi'_l$, and each $\Phi_m = \bigcirc\Phi^1_m \vee \Phi^2_m$, with Φ^2_m being a UnCQ. Moreover,

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

Algorithm 1 $\mathsf{AnswerTCQ}(\mathcal{K}, \Phi, i)$ for $\mathcal{K}$ of length $n+1$

1: **if** $i > n$ **then return** $\emptyset$ **end if**
2: $C := \mathsf{Ind}(\mathcal{K})^{|\mathsf{AVar}(\Phi)|}$
3: $\Phi' := \mathsf{Transform}(\Phi)$ // Φ' in CNF
4: **for all** conjuncts Ψ in Φ' **do**
5: **if** $\Psi = \bigcirc\Psi'$ **then**
6: $D := \mathsf{AnswerTCQ}(\mathcal{K}, \Psi, i+1)$
7: **else if** $\Psi = \Psi_1 \vee \bigcirc\Psi_2$ or $\Psi = \bigcirc\Psi_2 \vee \Psi_1$ **then**
8: $D := \mathsf{AnswerUnCQ}(\mathcal{K}_i, \Psi_1) \cup$ // Ψ_1 non-temporal
$\mathsf{AnswerTCQ}(\mathcal{K}, \Psi_2, i+1)$
9: **else**
10: $D := \mathsf{AnswerUnCQ}(\mathcal{K}_i, \Psi)$ // Ψ non-temporal
11: **end if**
12: $C := C \cap D$
13: **end for**
14: **return** C

Here, for a TKB $\mathcal{K} = (\mathcal{O}, (\mathcal{D}_{i\in\{0,\dots,n\}}))$, we define $\mathcal{K}_{\geq j} := (\mathcal{O}, (\mathcal{D}_{i\in\{j,\dots,n\}}))$. Note that if $j > n$, $\mathcal{K}_{\geq j}$ has no data and $\mathsf{cert}_{\mathcal{K}_{\geq j}}(\Phi) = \emptyset$ for any TCQ Φ. Recall as well that $\mathcal{K}_j := (\mathcal{O}, \mathcal{D}_j)$. Lemma 2 and 3 are proven in the supplementary material (Westhofen, Jung, and Neider 2024).

Algorithm

We now derive our procedure for answering TCQs, depicted as Algorithm 1, by leveraging Lemma 3 in a straightforward way. Recall that Algorithm 1 w.l.o.g. requires Φ to be in NNF. First, the query rewriting is performed (Line 3), resulting in a TCQ Φ' for which we iterate over all conjuncts Ψ. Per Lemma 3, each Ψ is either a UnCQ (where we determine $\mathsf{cert}_{\mathcal{K}_i}(\Psi)$), a $\bigcirc$ formula (where we advance to the next time point), or a disjunction where one disjunct is non-temporal and the other is a $\bigcirc$ formula (in which case we can combine the answers to both disjuncts). Finally, the set of certain answers is computed by $\mathsf{AnswerTCQ}(\mathcal{K}, \Phi, 0) = \mathsf{cert}_{\mathcal{K}}(\Phi)$. Correctness of this (Theorem 1) is shown in the supplementary material and based mainly on Lemmas 1 and 3.

Theorem 1. *For a TKB $\mathcal{K}$ and TCQ Φ it holds that $\mathsf{AnswerTCQ}(\mathcal{K}, \Phi, 0) = \mathsf{cert}_{\mathcal{K}}(\Phi)$.*

For illustration of our procedure, we use the example of Section 1 with a global type constraint: $\Phi = \Box\mathtt{Human}(x) \wedge \Diamond\mathtt{Pedestrian}(x) \equiv \mathsf{false}\,\mathcal{R}\mathtt{Human}(x) \wedge \mathsf{true}\,\mathcal{U}\mathtt{Pedestrian}(x)$. Over the example TKB $\mathcal{K}_{ex}$, which we now simply denote by $\mathcal{K}$, $\mathsf{cert}_{\mathcal{K}}(\Phi) = \{\mathtt{Jon}, \mathtt{Jane}\}$.

Our procedure applies Rules 3 and 4 on $\mathsf{true}\,\mathcal{U}\,\mathtt{Pedestrian}(x)$ and $\mathsf{false}\,\mathcal{R}\mathtt{Human}(x)$, continues to resolve the resulting disjunctions by Rule 2, and finally uses Rule 5 to eliminate $\CIRCLE$. For brevity, we omit intermediate formulae. The result is $\Phi' = \mathtt{Human}(x) \wedge (\bigcirc(\mathsf{false}\,\mathcal{R}\mathtt{Human}(x)) \vee \mathsf{last}) \wedge (\bigcirc(\mathsf{true}\,\mathcal{U}\mathtt{Pedestrian}(x)) \vee \mathtt{Pedestrian}(x))$. We can now use AnswerUnCQ, as the temporal disjunctions $\bigcirc(\mathsf{false}\,\mathcal{R}\mathtt{Human}(x)) \vee \mathsf{last}$ and $\bigcirc(\mathsf{true}\,\mathcal{U}\,\mathtt{Pedestrian}(x)) \vee \mathtt{Pedestrian}(x)$ adhere to $\bigcirc\Phi_1 \vee \Phi_2$ with Φ_2 non-temporal. We now compute $\mathsf{cert}_{\mathcal{K}_0}(\mathtt{Human}(x)) = \mathsf{cert}_{\mathcal{K}_0}(\mathtt{Pedestrian}(x)) = \{\mathtt{Jane}, \mathtt{Jon}\}$, and $\mathsf{cert}_{\mathcal{K}_0}(\mathsf{last}) = \emptyset$. Advancing to $\mathcal{K}_1$ recur-

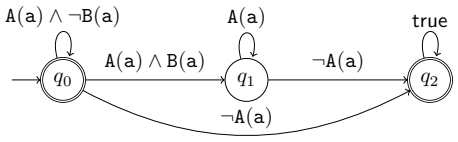

Figure 2: FA for checking satisfiability of $\neg(\Box \mathtt{A}(\mathtt{a}) \wedge \Diamond \mathtt{B}(\mathtt{a}))$.

sively descends into the $\bigcirc$ subformulae of Φ', i.e., $\mathsf{false}\,\mathcal{R}\mathtt{Human}(x)$ and $\mathsf{true}\,\mathcal{U}\mathtt{Pedestrian}(x)$, for which the above steps are repeated, with the difference that $\mathsf{cert}_{\mathcal{K}_1}(\mathtt{Pedestrian}(x)) = \emptyset$. The inductive combination in Line 12 leads to the result $\mathsf{cert}_{\mathcal{K}}(\Phi) = \{\mathtt{Jane}, \mathtt{Joe}\}$.

Discussion

We start discussing our approach by studying its complexity (a full proof is given in the supplementary material).

Theorem 2. *For a TCQ Φ and an $\mathcal{ALC}$-TKB $\mathcal{K}$,* $\mathsf{AnswerTCQ}(\mathcal{K}, \Phi, 0)$ *runs in exponential time.*

Proof (Sketch). $\mathsf{Transform}(\Phi)$ creates, at each time point, at worst an exponential formula (due to Rule 2). For each of the exponentially many conjuncts Ψ, the next time point is examined or $\mathsf{AnswerUnCQ}$ is called on Ψ, with Ψ being linear in the size of Φ. Thus, at each time, we do exponentially many checks of linearly sized UnCQs. As $\mathsf{AnswerUnCQ}$ runs in exponential time for $\mathcal{ALC}$ (Baader, Borgwardt, and Lippmann 2013), $\mathsf{AnswerTCQ}$ is exponential as well. □

Since TCQ entailment over $\mathcal{ALC}$ is ExpTime-complete (Baader, Borgwardt, and Lippmann 2013; Westhofen et al. 2024a), Algorithm 1 is optimal for $\mathcal{ALC}$. The optimality also holds for more expressive DLs such as $\mathcal{ALCI}$, where query entailment is 2ExpTime-complete (Lutz 2007) and hence $\mathsf{AnswerTCQ}(\mathcal{K}, \Phi, 0)$ runs in double-exponential time. Our approach can be adapted to Horn DLs as well, since they allow to simply combine answers to disjuncts. In this case, a conjunction of disjunctions cannot be assumed, and one has to instead implement the recursive semantics of Definition 4.

While the automata-based approach is theoretically optimal as well, there is a major difference: In some cases where our approach relies solely on highly optimized CQ answering, the FA-based approach checks unnecessary UCQs.

Consider, for example, the TCQ $\Box \mathtt{A}(\mathtt{a}) \wedge \Diamond \mathtt{B}(\mathtt{a})$ over $\mathcal{K} = (\emptyset, (\{\mathtt{A}(\mathtt{a})\}, \{\mathtt{A}(\mathtt{a}), \mathtt{B}(\mathtt{a})\}))$. The corresponding FA of the negated TCQ is given in Figure 2. For $t = 0$ and q_0, without UCQ answering and even with the CQ answering optimizations sketched by Westhofen et al., we can only assert the edge to q_2 to not hold. Yet, at least one of the edges to q_0 or q_1 must be satisfied, but we cannot just arbitrarily choose one, as they induce a different behavior at $t = 1$. Since both edges require checking satisfiability of conjunctions of CQs, we encounter unnecessary UCQ checks – recall that Algorithm 1 uses only CQ answering on this TCQ.

We believe this behavior to originate from multiple indirections of the FA-based approach: It checks unsatisfiability of the negated TCQ by constructing its FA and then satisfiability of its edges during traversal. It appears that UCQ checks arise due to the FA construction not propagating the negation of the overall TCQ onto its edges but rather encoding it in the final states. It thus does not resolve entailment of conjunctions of CQs in the original TCQ to satisfiability of disjunctions of CQs in the FA. Rather, it encodes them as satisfiability checks of conjunctions of possibly negated CQs which is, unfortunately, equivalent to entailment of UCQs. While this has no implications on theoretical complexity, CQ answering is highly optimizable, e.g., through 'atom-by-atom' examination while examining the completion graph from the initial consistency check (Sirin 2006). We next show empirical implications of this observation.

5 Evaluation

We implemented Algorithm 1 into Openllet, the same reasoner that was used in Westhofen et al. (2024a). Thus, the systems operate on the same CQ and UCQ engines, where the latter is a custom implementation on top of Openllet. Both systems assume finite data and are restricted to tree-shaped queries, i.e., in the graph induced by the query's roles, no undirected cycles exist and each node has at most one incoming edge. Moreover, we complement Algorithm 1 by formula simplification, e.g., $\Phi \wedge \mathsf{false} \rightsquigarrow \mathsf{false}$, and, for input language equivalence w.r.t. Westhofen et al. (2024a), add metric operators by regarding them during rewriting.

For the TKBs, we rely on the only two existing benchmark sets for temporal querying over expressive DLs known to us, both publicly available and provided by Westhofen et al. (2024a, 2024b). The first family, called the *traffic ontology benchmark (TOBM)* (Westhofen et al. 2024a) models pedestrians, bicyclists, and vehicles over 200 time points on an X- and a T-crossing with four TCQs ($\Phi_1, \Phi_2, \Phi_3, \Phi_4$) and a large $\mathcal{SRIQ}^{(\mathcal{D})}$ ontology of 676 concepts and 213 roles. TOBM allows to linearly scale the number of individuals by a factor N, which we set to $N \in \{1, \ldots, 5\}$. For $N = 5$, this leads to roughly 15 642 assertions on the X-crossing and 5 880 assertions on the T-crossing per time point. The second benchmark category is concerned with a single query Φ_1 for granting right of way to two wheelers and provides TKBs with just three time points (Westhofen et al. 2024b). The TKB is again scalable, for which we selected $N \in \{1, \ldots, 30\}$, where $N = 30$ has, on average, 115 assertions per time point. Overall, we assembled 70 benchmarks, i.e., TKB-TCQ combinations. All TCQs are compatible with our input language. Code, data, and reproducibility instructions are provided online (Westhofen 2024).

We ran each system once per benchmark (as determinism of both implementations makes deviations negligible) on a Windows 10 machine with an Intel Core i9-13900K, 64 GB RAM, and a ten hours time limit per run. We measured wall clock times for query execution without loading of TKBs (called 'run time' from here on). Moreover, we collected information on the number of UCQ entailment checks and calls to the CQ answering engine, as well as run time spent in both. Our main results are depicted in Figure 3 and indicate that performance is improved on *all benchmarks*. For three

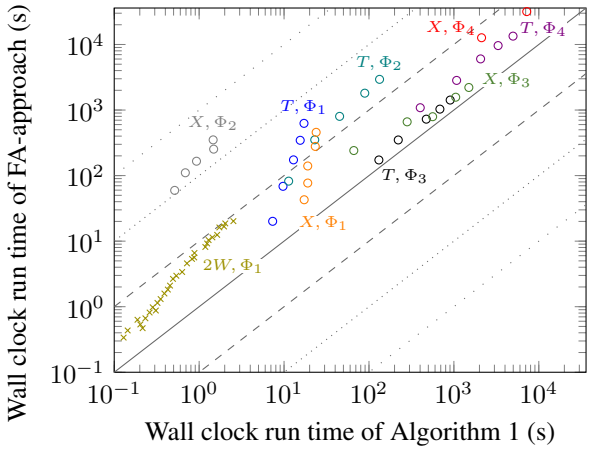

Figure 3: Log-scaled scatter plot of the wall clock run times for the FA-based implementation of Westhofen et al. (2024b) against our Algorithm 1 on the traffic ontology (X and T, circles) and the two-wheeler benchmarks (2W, crosses). Colors indicate the membership of scaled benchmarks to the same type. Lines are orders of magnitudes.

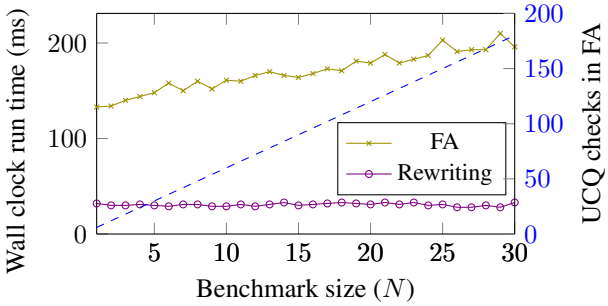

Figure 4: Wall clock run times of Algorithm 1 vs. the system of Westhofen et al. (2024b) and its number of UCQ entailment checks for the two-wheeler benchmark on a TCQ of the form $\Box\mathtt{A}(x) \wedge \Diamond\mathtt{B}(x)$. Algorithm 1 uses no UCQ checks.

benchmarks, the FA-based implementation exceeds the time limit, whereas our approach is able to answer the queries in 3.52, 5.72, and 7.88 hours. On all other benchmarks, we achieve an average speedup of 19.09, that is, over an order of magnitude, with a standard deviation of 45.79.

We earlier conjectured that this is due to a more effective use of non-temporal query answering. This behavior might be specific to the employed (U)CQ engines (which both systems rely on). As we are not aware of other UCQ engines over expressive DLs, we restrict our analysis to the engines available in Openllet. However, since UCQ answering is practically more challenging than CQ answering, we conjecture that the number of calls to query engines is a reasonably generalizable proxy. We hence show the number of UCQ entailment checks, calls to the CQ engine, and their run times in Table 1. In our data, often, UCQ answering takes up most of the time in the FA-based approach (mainly as it cannot rely on the same optimizations as CQ answering), indicating that decreasing the number of UCQ entailment checks promises large benefits. In fact, in four cases, we eliminated UCQ checks altogether. On six of the nine benchmarks, run times are improved largely due to reducing UCQ answering times. There are, however, three exceptions: On Φ_2 on the X-crossing and Φ_3 in both TKBs, the CQ engine takes up most or all time, and hence the effect of reducing UCQ answering is small or non-existent. Still, our approach offers a significant speedup on those benchmarks as well by decreasing the time spent on CQ answering.

While Table 1 shows that there are queries for which the FA-based approach checks UCQs excessively on *some* TKBs, we previously argued that, for some queries, UCQs are actually not required on *any* TKB. An example is the TCQ $\Box\mathtt{A}(\mathtt{a}) \wedge \Diamond\mathtt{B}(\mathtt{a})$, which Algorithm 1 solves with CQ answering only – we claimed that the FA-based approach uses UCQ answering regardless. To substantiate our claim, we executed this TCQ on the two-wheeler TKBs. As Figure 4 shows, Algorithm 1 does not perform UCQ checks, whereas the implementation of Westhofen et al. exhibits higher run times due to a linearly increasing number of UCQ checks.

6 An Efficiently Decidable TCQ Fragment

Our empirical data motivate that it is desirable to rely solely on CQs for TCQ answering (as opposed to inefficient UCQ answering). In general, this is not possible for expressive DLs. Yet, there are some queries which can be answered by using only CQs, e.g., $\Box\mathtt{A}(x) \wedge \Diamond\mathtt{B}(x)$ as demonstrated previously. In fact, this is possible for a range of TCQs, including 'safety' ($\Box\varphi_1$), 'liveness' ($\Diamond\varphi_1$), 'one-off until' ($(\bigcirc\varphi_1)\,\mathcal{U}\,\varphi_2$), and conjunctions thereof, for CQs φ_1, φ_2. However, note that seemingly benign queries like $\Phi_b = \Diamond(\mathtt{A}(\mathtt{a}) \wedge \bigcirc\mathtt{B}(\mathtt{a}))$ must be checked by UCQ answering. Consider $\mathcal{K}_b = (\{\top \sqsubseteq \mathtt{A} \sqcup \mathtt{B}\}, (\{\mathtt{A}(\mathtt{a})\}, \emptyset, \{\mathtt{B}(\mathtt{a})\}))$ for which, counter-intuitively, $\mathcal{K}_b \models \Phi_b$. Essentially, $\mathtt{A}(\mathtt{a}) \vee \mathtt{B}(\mathtt{a})$ has to be checked at the second time point originating from combining $\Diamond$ and $\bigcirc$. While it is easy to see that a reduction to CQ answering is possible for non-trivial queries, Φ_b highlights that this does not have to be immediately obvious.

Ideally, users are able to automatically and efficiently check this property for a given query before its execution. To achieve this, we now formally characterize a subset of all TCQs that can be answered by CQs only and start with a generic characterization of such TCQs in Definition 7.

Definition 7 ($\mathrm{TCQ}_{\mathrm{CQ}}$). *A TCQ Φ is in $\mathrm{TCQ}_{\mathrm{CQ}}$ if there is an algorithm that answers Φ over arbitrary TKBs $\mathcal{K}$ and whose only access to the data is provided by a CQ answering oracle for the non-temporal KBs $\mathcal{K}_j$.*

Note that Definition 7 is a semantic one, and it does not directly give us a procedure to decide membership. It is thus of interest to identify a syntactic fragment $\mathrm{TCQ}_{\mathrm{CQ}^*}$ of $\mathrm{TCQ}_{\mathrm{CQ}}$ for which membership can be efficiently decided, e.g., to guide the user into efficient fragments during specification. First, we define $1 + S \coloneqq \{1 + s \mid s \in S\}$ and $S + T \coloneqq \{s + t \mid s \in S, t \in T\}$ for sets $S, T \subseteq \mathbb{N}$. For a TCQ Φ, we define $t(\Phi)$ – the time points it refers to – as

Benchmark		FA					Rewriting					Run Time Saved	Saved by UCQ (%)
TKB	Φ	UCQs	t_{UCQ}	CQs	t_{CQ}	t	UCQs	t_{UCQ}	CQs	t_{CQ}	t		
T-Cr.	Φ_1	94 720	590.61	593	35.70	626.31	0	0	397	17.03	17.17	609.14	96.96
	Φ_2	59 061	2 407.80	1 203	530.42	2 938.23	7 037	92.42	602	40.58	133.23	2 805.00	82.55
	Φ_3	17 299	115.29	603	1 304.76	1 420.05	0	0	402	904.00	904.20	515.85	22.35
	Φ_4	962 613	12 812.79	1 203	595.79	13 408.60	136 732	4 793.19	602	177.03	4 970.80	8 437.80	95.04
X-Cr.	Φ_1	27 824	392.27	593	64.74	457.02	0	0	397	23.77	23.88	433.14	90.56
	Φ_2	0	0	1 203	350.72	350.73	0	0	1	1.20	1.46	349.27	0.00
	Φ_3	16 780	329.33	603	1 879.35	2 208.70	0	0	402	1 500.09	1 500.33	708.37	46.49
	Φ_4	1 697 010	31 606.45	1 203	126.68	31 733.18	204 800	7 188.71	602	17.58	7 206.80	24 526.38	99.56
2-Wh.	Φ_1	5 232	17.51	45	2.66	20.19	357	0.27	8	2.21	2.53	17.66	97.62

Table 1: Number of UCQ entailment and CQ answering calls, their run times, and overall run times (in seconds) for the system of Westhofen et al. (2024b) ('FA') and Algorithm 1 ('Rewriting') and the largest benchmark of each benchmark family where no timeout occurred, together with the percentage of the overall saved run time that is due to savings in UCQ entailment checks.

- $t(\mathsf{true}) = t(\mathsf{false}) = t(\mathsf{last}) = \emptyset$,
- $t(\varphi) = \{0\}$ for any CQ φ,
- $t(\Phi_1 \vee \Phi_2) = t(\Phi_1 \wedge \Phi_2) = t(\Phi_1) \cup t(\Phi_2)$,
- $t(\bullet\!\!\!\bigcirc \Phi') = t(\bigcirc \Phi') = 1 + t(\Phi')$, and
- $t(\Phi_1 \,\mathcal{R}\, \Phi_2) = t(\Phi_1 \,\mathcal{U}\, \Phi_2) = \mathbb{N} + (t(\Phi_1) \cup t(\Phi_2))$.

Definition 8 (TCQ_{CQ^*})**.** *Φ is in TCQ_{CQ^*} if it does not contain negations and for all subformulae of the form*

1. *$\Phi_1 \,\mathcal{U}\, \Phi_2$ we have $t(\Phi_2) \cap (t(\Phi_1) \cup (1 + t(\Phi_1 \,\mathcal{U}\, \Phi_2))) = \emptyset$,*
2. *$\Phi_1 \,\mathcal{R}\, \Phi_2$ we have $t(\Phi_1) \cap (1 + t(\Phi_1 \,\mathcal{R}\, \Phi_2)) = \emptyset$,*
3. *$\Phi_1 \vee \Phi_2$ we have $t(\Phi_1) \cap t(\Phi_2) = \emptyset$.*

Before showing that TCQ_{CQ^*} is in fact a fragment of TCQ_{CQ}, we first require a supplementary statement.

Lemma 4. *Let $\Phi \in \text{TCQ}_{\text{CQ}^*}$ and $\Psi = \mathsf{Transform}(\Phi)$. Any subformula of Ψ is again in TCQ_{CQ^*}.*

Proof. The statement follows from the facts that any subformula of some formula in TCQ_{CQ^*} is also in TCQ_{CQ^*} (as per Definition 8) and that TCQ_{CQ^*} is closed under $\mathsf{Transform}$ (which is proven in the supplementary material). □

We can now prove Theorem 3 – stating the correctness of the syntactic TCQ_{CQ} fragment – by showing that for $\Phi \in \text{TCQ}_{\text{CQ}^*}$, $\mathsf{AnswerTCQ}(\Phi, \mathcal{K}, i)$ uses only CQ answering for KB reasoning (i.e., when applying $\mathsf{AnswerUnCQ}$).

Theorem 3. *$\text{TCQ}_{\text{CQ}^*} \subsetneq \text{TCQ}_{\text{CQ}}$.*

Proof. First, $\mathsf{AnswerTCQ}(\mathcal{K}, \Phi, i)$ encounters only CQs in $\mathsf{AnswerUnCQ}$ if we do not consider recursive calls: As per Lemma 4, TCQ_{CQ^*} is closed under $\mathsf{Transform}$ and thus $\mathsf{Transform}(\Phi) \in \text{TCQ}_{\text{CQ}^*}$. By Constraint 3 in Definition 8, $\mathsf{Transform}(\Phi)$ cannot contain an atemporal disjunction of two CQs and Lines 8 and 10 only encounter CQs. Due to Lemma 4, the inputs to $\mathsf{AnswerTCQ}$ in Lines 6 and 8 satisfy again the assumption that the query is in TCQ_{CQ^*} and thus the above argument is also applicable to the recursive calls at time $i + 1$. Therefore, $\text{TCQ}_{\text{CQ}^*} \subseteq \text{TCQ}_{\text{CQ}}$.

Finally, it is easy to see that TCQ_{CQ^*} does not cover the whole class of TCQ_{CQ}, e.g., $\Diamond(\varphi_1 \,\mathcal{U}\, \varphi_2)$ for two CQs φ_1, φ_2 can be answered using a CQ engine (as it is equivalent to $\Diamond\varphi_2$) but is not in TCQ_{CQ^*}. Hence, $\text{TCQ}_{\text{CQ}^*} \subsetneq \text{TCQ}_{\text{CQ}}$. □

For being of practical use in application scenarios (such as guiding users towards efficient query fragments) we require that syntactic membership in TCQ_{CQ^*} can be efficiently decided before answering (which might be exponential).

Theorem 4. *Syntactic membership in TCQ_{CQ^*}, i.e. $\Phi \in \text{TCQ}_{\text{CQ}^*}$, is decidable in polynomial time.*

Theorem 4 is proven in the supplementary material. Intuitively, computing t requires one traversal of the formula's syntax tree with operations that take polynomial time on the specific shapes of the induced sets. The constraints of Definition 8 can be checked in polynomial time as well.

7 Conclusion

We presented a novel approach for answering TCQs over temporal data and ontologies in expressive Description Logics by a query rewriting approach. Experimental comparison with a state of the art tool exhibited speed-ups by up to two orders of magnitude.

In the future, we would like to extend our approach to allowing past temporal operators like 'at the previous time point'. We also aim to deepen our understanding of the dependency on UCQs: First, is our novel approach provably guaranteed to need at most as many accesses to the UCQ engine as *any* approach based on compilation into automata? Second, can we (efficiently) decide TCQ_{CQ}? Third, can we extend TCQ_{CQ^*}, e.g., with a restricted form of negation, while keeping its favourable properties?

An orthogonal direction is to study stronger forms of rewritings where not only the temporal component of the query is rewritten, but the full TCQ. A good starting point for doing so might be temporal KBs with an ontology formulated in some Horn description logic. Possible target languages are first-order temporal logic or temporal Datalog (Artale et al. 2022). Such rewritings are certainly not always possible, but where they are, performance could be significantly improved.

Acknowledgements

This work was partially funded by the German Research Foundation (DFG) under grant no. 459419731 and the German Federal Ministry for Education and Research (BMBF) as part of 'MANNHEIM-AutoDevSafeOps' under reference no. 01IS22087C.

References

Artale, A.; and Franconi, E. 2000. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence*, 30(1-4): 171–210.

Artale, A.; Kontchakov, R.; Kovtunova, A.; Ryzhikov, V.; Wolter, F.; and Zakharyaschev, M. 2017. Ontology-Mediated Query Answering over Temporal Data: A Survey (Invited Talk). In Schewe, S.; Schneider, T.; and Wijsen, J., eds., *24th International Symposium on Temporal Representation and Reasoning, TIME 2017, Mons, Belgium, October 16-18, 2017*, volume 90 of *LIPIcs*, 1:1–1:37. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Artale, A.; Kontchakov, R.; Kovtunova, A.; Ryzhikov, V.; Wolter, F.; and Zakharyaschev, M. 2022. First-Order Rewritability and Complexity of Two-Dimensional Temporal Ontology-Mediated Queries. *Journal of Artificial Intelligence Research*, 75: 1223–1291.

Artale, A.; Kontchakov, R.; Ryzhikov, V.; and Zakharyaschev, M. 2014. A Cookbook for Temporal Conceptual Data Modelling with Description Logics. *ACM Transactions on Computational Logic*, 15(3): 25:1–25:50.

Artale, A.; Kontchakov, R.; Wolter, F.; and Zakharyaschev, M. 2013. Temporal Description Logic for Ontology-Based Data Access. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI 2013, Beijing, China, August 3-9, 2013*, 711–717. Palo Alto, USA: IJCAI/AAAI.

Baader, F.; Borgwardt, S.; and Lippmann, M. 2013. Temporalizing Ontology-Based Data Access. In Bonacina, M. P., ed., *24th International Conference on Automated Deduction, CADE-24, Lake Placid, NY, USA, June 9-14, 2013*, volume 7898 of *Lecture Notes in Computer Science*, 330–344. Berlin, Germany: Springer Nature.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2007. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge, USA: Cambridge University Press, 2nd edition. ISBN 978-0-511-71178-7.

Baier, C.; and Katoen, J. 2008. *Principles of model checking*. Cambridge, USA: MIT Press. ISBN 978-0-262-02649-9.

Borgwardt, S.; Lippmann, M.; and Thost, V. 2013. Temporal Query Answering in the Description Logic DL-Lite. In Fontaine, P.; Ringeissen, C.; and Schmidt, R. A., eds., *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*, 165–180. Berlin, Germany: Springer Nature. ISBN 978-3-642-40885-4.

Borgwardt, S.; Lippmann, M.; and Thost, V. 2015. Temporalizing rewritable query languages over knowledge bases. *Journal of Web Semantics*, 33: 50–70.

Borgwardt, S.; and Thost, V. 2015. Temporal Query Answering in the Description Logic EL. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, 2819–2825. Palo Alto, USA: AAAI Press. ISBN 9781577357384.

Calvanese, D.; Okulmus, C.; Ortiz, M.; and Simkus, M. 2023. On the Way to Temporal OBDA Systems (short paper). In Kimelfeld, B.; Martinez, M. V.; and Angles, R., eds., *Proceedings of the 15th Alberto Mendelzon International Workshop on Foundations of Data Management, AMW 2023, Santiago de Chile, Chile, May 22-26, 2023*, volume 3409 of *CEUR Workshop Proceedings*. Aachen, Germany: CEUR-WS.org.

De Giacomo, G.; and Vardi, M. Y. 2013. Linear temporal logic and linear dynamic logic on finite traces. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI 2013, Beijing, China, August 3-9, 2013*, 854–860. Palo Alto, USA: IJCAI/AAAI. ISBN 9781577356332.

Gutiérrez-Basulto, V.; Jung, J. C.; and Kontchakov, R. 2016. Temporalized EL Ontologies for Accessing Temporal Data: Complexity of Atomic Queries. In Kambhampati, S., ed., *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, July 9-15, 2016*, 1102–1108. Palo Alto, USA: IJCAI/AAAI Press.

Kalayci, E. G.; Xiao, G.; Ryzhikov, V.; Kalayci, T. E.; and Calvanese, D. 2018. Ontop-temporal: A Tool for Ontology-based Query Answering over Temporal Data. In Cuzzocrea, A.; Allan, J.; Paton, N. W.; Srivastava, D.; Agrawal, R.; Broder, A. Z.; Zaki, M. J.; Candan, K. S.; Labrinidis, A.; Schuster, A.; and Wang, H., eds., *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, 1927–1930. New York, USA: ACM Press.

Kusano, K. D.; Scanlon, J. M.; Chen, Y.-H.; McMurry, T. L.; Chen, R.; Gode, T.; and Victor, T. 2023. Comparison of Waymo Rider-only crash data to human benchmarks at 7.1 million miles. *CoRR*, abs/2312.12675.

Li, J.; Vardi, M. Y.; and Rozier, K. Y. 2019. Satisfiability checking for mission-time LTL. In Dillig, I.; and Tasiran, S., eds., *Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part II*, volume 11562 of *Lecture Notes in Computer Science*, 3–22. Berlin, Germany: Springer Nature. ISBN 978-3-030-25543-5.

Lutz, C. 2007. Inverse Roles Make Conjunctive Queries Hard. In Calvanese, D.; Franconi, E.; Haarslev, V.; Lembo, D.; Motik, B.; Turhan, A.; and Tessaris, S., eds., *Proceedings of the 2007 International Workshop on Description Logics, DL 2007, Brixen-Bressanone, near Bozen-Bolzano, Italy, June 8-10, 2007*, volume 250 of *CEUR Workshop Proceedings*. Aachen, Germany: CEUR-WS.org.

Lutz, C.; Wolter, F.; and Zakharyaschev, M. 2008. Temporal Description Logics: A Survey. In Demri, S.; and Jensen, C. S., eds., *15th International Symposium on Temporal Representation and Reasoning, TIME 2008, Université*

du Québec à Montréal, Canada, June 16-18, 2008, 3–14. New York, USA: IEEE Computer Society.

Parsia, B.; Matentzoglu, N.; Gonçalves, R. S.; Glimm, B.; and Steigmiller, A. 2017. The OWL reasoner evaluation (ORE) 2015 competition report. *Journal of Automated Reasoning*, 59(4): 455–482.

Sirin, E. 2006. *Combining Description Logic Reasoning with AI Planning for Composition of Web Services*. Ph.D. thesis, University of Maryland, College Park, MD, USA.

Sirin, E.; Parsia, B.; Grau, B. C.; Kalyanpur, A.; and Katz, Y. 2007. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2): 51–53.

Thost, V.; Holste, J.; and Özçep, Ö. L. 2015. On Implementing Temporal Query Answering in DL-Lite (extended abstract). In Calvanese, D.; and Konev, B., eds., *Proceedings of the 28th International Workshop on Description Logics, DL 2015, Athens, Greece, June 7-10, 2015*, volume 1350 of *CEUR Workshop Proceedings*. Aachen, Germany: CEUR-WS.org.

Wang, D.; Hu, P.; Wałega, P. A.; and Grau, B. C. 2022. MeTeoR: practical reasoning in Datalog with metric temporal operators. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Virtual Event, February 22 - March 1, 2022*, volume 36, 5906–5913. Palo Alto, USA: AAAI Press.

Westhofen, L. 2024. Experiments for 'Temporal Conjunctive Query Answering via Rewriting'. https://doi.org/10.5281/zenodo.14412131. Accessed: 2025-01-07.

Westhofen, L.; Jung, J. C.; and Neider, D. 2024. Supplementary Material to 'Temporal Conjunctive Query Answering via Rewriting'. https://doi.org/10.5281/zenodo.14516502. Accessed: 2025-01-07.

Westhofen, L.; Neurohr, C.; Jung, J. C.; and Neider, D. 2024a. Answering Temporal Conjunctive Queries over Description Logic Ontologies for Situation Recognition in Complex Operational Domains. In Finkbeiner, B.; and Kovács, L., eds., *Tools and Algorithms for the Construction and Analysis of Systems - 30th International Conference, TACAS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part I*, 167–187. Berlin, Germany: Springer Nature. ISBN 978-3-031-57246-3.

Westhofen, L.; Neurohr, C.; Jung, J. C.; and Neider, D. 2024b. Topllet: An Optimized Engine for Answering Metric Temporal Conjunctive Queries. In Benz, N.; Gopinath, D.; and Shi, N., eds., *NASA Formal Methods - 16th International Symposium, NFM 2024, Moffett Field, CA, USA, June 4-6, 2024, Proceedings*, volume 14627 of *Lecture Notes in Computer Science*, 314–321. Berlin, Germany: Springer Nature. ISBN 978-3-031-60697-7.