Practical Querying of Temporal Data via OWL 2 QL and SQL:2011

Szymon Klarman

Centre for Artificial Intelligence Research,
CSIR Meraka and University of KwaZulu-Natal, South Africa
sklarman@csir.co.za

Abstract

We develop a practical approach to querying temporal data stored in temporal SQL:2011 databases through the semantic layer of OWL 2 QL ontologies. An interval-based *temporal query language* (TQL), which we propose for this task, is defined via naturally characterizable combinations of temporal logic with conjunctive queries. This foundation warrants well-defined semantics and formal properties of TQL querying. In particular, we show that under certain mild restrictions the data complexity of query answering remains in AC^0, i.e., as in the usual, nontemporal case. On the practical side, TQL is tailored specifically to offer maximum expressivity while preserving the possibility of reusing standard first-order rewriting techniques and tools for OWL 2 QL.

1 Introduction

Ontology-based data access (OBDA) is a popular paradigm of managing information, which combines the data storage and querying capabilities offered by relational database management systems (RDBMSs) with semantically enhanced view on the data provided by ontologies. OWL 2 QL is an OWL 2 profile, based on the DL-*Lite* family of Description Logics, designed specifically to support optimally balanced OBDA [6]. As large portions of data available through the Web are in fact still hosted in traditional relational datastores, OBDA provides a crucial channel for accessing this data from the level of the Semantic Web applications. With this work, we aim to establish an analogical to OBDA interface between the semantic layer of OWL 2 QL and temporal data stored and managed using SQL:2011, the most recent standardization of the SQL query language, which provides unprecedented support for validity time of data [11]. In this way we hope to define a convenient framework for reasoning with temporal semantic data — the problem gaining ever more attention within the Semantic Web community.

The *temporal query language* (TQL), which we propose for this task, is based on naturally characterizable combinations of temporal logic with conjunctive queries, identified in [10]. TQL is tailored specifically to offer maximum expressivity while preserving the possibility of reusing first-order rewriting techniques and tools, central to OBDA and developed specifically for the use with OWL 2 QL. While this technical compliance warrants the practicality of our approach, its well-defined logic foundations allow us to identify basic formal properties of TQL. In particular, we are able to demonstrate that under a mild restriction to finite time domains the data complexity of query answering remains in AC^0, i.e., as in the standard, nontemporal case. As the main result, we present a translation of TQL queries in the presence of OWL 2 QL ontologies to SQL:2011, which opens the way to efficient query answering by means of existing, commercially supported RDBMSs, such as IBM DB2 10.1, Oracle Database 11g Workspace Manager, or Teradata — all having adopted by now certain variants of the SQL:2011 standard[1].

[1]See `http://www.cs.arizona.edu/~rts/sql3.html` for an overview.

In this report, we first introduce TQL and define the query answering task in Section 3, and then, in Section 4, we present our results on TQL query rewriting and data complexity. The related work is discussed in Section 5 and the paper is concluded in Section 6.

2 Preliminaries

We start by recapping the foundations of OBDA and temporal data. Next, we motivate our research problem with an illustrative application scenario.

2.1 Ontology-based data access

OWL 2 QL is a profile of OWL 2 Web Ontology Language, based on the DL-*Lite* family of Description Logics (DLs) [6]. A DL vocabulary $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_R})$ consists of countably infinite sets of individual names ($\mathsf{N_I}$), concept names ($\mathsf{N_C}$) and role names ($\mathsf{N_R}$). The ABox $\mathcal{A}$ is a finite set of assertions $A(a)$ and $r(a,b)$, for $a, b \in \mathsf{N_I}$, $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$. The TBox $\mathcal{T}$ is a finite set of concept and role inclusions $B \sqsubseteq C$, $r \sqsubseteq s$, built using logical constructors allowed in OWL 2 QL.[2] The semantics is given in terms of DL interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, defined as usual [4]. An interpretation $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$, denoted as $\mathcal{I} \models \mathcal{T}, \mathcal{A}$, *iff* it satisfies every axiom in $\mathcal{T}$ and $\mathcal{A}$.

In OBDA, the instance data, represented as an ABox, is accessed via an ontology, given as a TBox, using a designated query language, such as most commonly considered in that context, the language of conjunctive queries [9]. Let $\mathsf{N_V}$ be a countably infinite set of variables. A *conjunctive query* (CQ) over a DL vocabulary Σ is a first-order formula $\exists \vec{y}.\varphi(\vec{x}, \vec{y})$, where $\vec{x}, \vec{y}$ are sequences of variables. The sequence $\vec{x}$ denotes the free (individual) variables in the query, while $\vec{y}$ the quantified ones. The formula φ is a conjunction of atoms over Σ of the form $A(u), r(u,v)$, where $u, v \in \mathsf{N_V} \cup \mathsf{N_I}$ are called terms. By $\mathsf{term}(q)$ we denote the set of all terms occurring in a CQ q and by $\mathsf{obj}(q)$ the set of all its free variables. We call q grounded whenever $\mathsf{obj}(q) = \emptyset$. A grounded CQ q is satisfied in $\mathcal{I}$ *iff* there exists a mapping $\mu : \mathsf{term}(q) \mapsto \Delta^{\mathcal{I}}$, with $\mu(a) = a^{\mathcal{I}}$ for every $a \in \mathsf{N_I}$, such that $\mu(u) \in A^{\mathcal{I}}$ and $(\mu(u), \mu(v)) \in r^{\mathcal{I}}$ for every $A(u)$ and $r(u,v)$ in q. Further, we say that q is entailed by $\mathcal{T}, \mathcal{A}$, denoted as $\mathcal{T}, \mathcal{A} \models q$ *iff* q is satisfied in every model of $\mathcal{T}, \mathcal{A}$. Whenever $\emptyset, \mathcal{A} \models q$ we also write $\mathcal{A} \models q$. An *answer* to q is a mapping $\sigma : \mathsf{obj(q)} \mapsto \mathsf{N_I}$. By $\sigma(q)$ we denote the result of uniformly substituting every occurrence of x in q with $\sigma(x)$, for every $x \in \mathsf{obj(q)}$. An answer σ is called *certain* over $\mathcal{T}$, $\mathcal{A}$ *iff* $\sigma(q)$ is entailed by $\mathcal{T}, \mathcal{A}$. By $\mathcal{Q}_{\Sigma}$ we denote the class of all CQs over the vocabulary Σ.

A prominent property of CQs is their *first-order* (FO) *rewritability* in OWL 2 QL.

Definition 1 (FO rewritability [6])**.** *For every CQ $q \in \mathcal{Q}_{\Sigma}$ and a TBox $\mathcal{T}$, there exists a FO formula $q^{\mathcal{T}}$, called the* FO rewrting of q in $\mathcal{T}$*, such that for every ABox $\mathcal{A}$ and answer σ to q, it holds that:*

$$\mathcal{T},\, \mathcal{A} \models \sigma(q) \ \text{ iff } \ db(\mathcal{A}) \Vdash \sigma(q^{\mathcal{T}}),$$

where $\Vdash$ is the FO satisfaction relation and $db(\mathcal{A})$ denotes $\mathcal{A}$ considered as a database/FO interpretation, i.e., a structure $(\mathsf{N_I}, \cdot^{\mathcal{D}})$, where $\mathsf{N_I}$ serves as the interpretation domain and $\cdot^{\mathcal{D}}$ is an interpretation function, defined as:

- $A^{\mathcal{D}} = \{a \in \mathsf{N_I} \mid A(a) \in \mathcal{A}\}$*, for every* $A \in \mathsf{N_C}$,
- $r^{\mathcal{D}} = \{(a,b) \in \mathsf{N_I} \times \mathsf{N_I} \mid r(a,b) \in \mathcal{A}\}$*, for every* $r \in \mathsf{N_R}$.

[2]See `http://www.w3.org/TR/owl2-profiles/#OWL_2_QL` for full details.

FO rewritability is particularly significant from the practical perspective. It implies that answering CQs in OWL 2 QL can be effectively performed in existing RDBMSs via a translation to SQL, as the ontological component can be always compiled out in the query rewritting, without loss of soundness or completeness. As a theoretical corollary, it follows also that the data complexity of CQ answering in this setup is AC^0, as in the first-order logic (FOL).

In what follows, we focus exclusively on OWL 2 QL TBoxes, even though some of the presented results should clearly transfer to other fragments of DLs enjoying FO rewritability. We hence implicitly assume that every TBox or ontology mentioned in the remainder of this report is expressed in OWL 2 QL.

2.2 Temporal databases

SQL:2011 takes a minimal, regulative approach to extending the relational data model with the temporal dimension [11]. It accounts for two extra columns included in database's tables, designated for storing the beginning and the end time of the validity periods of respective records. Formally, such time-stamping mechanism is based on intervals over linear time domains.

Definition 2 (Time domain, time intervals). *A* time domain *is a tuple* $\mathfrak{T} = (T, <)$*, where* T *is a nonempty set of elements called* time points *and* $<$ *is an irreflexive, linear ordering on* T*. A* time interval $\tau = [\tau^-, \tau^+]$ *over* $\mathfrak{T}$ *is a set of time points* $\{t \in T \mid \tau^- \leq t \leq \tau^+\}$*, where* $\tau^-, \tau^+ \in T$*, such that* $\tau^- \leq \tau^+$*, are called the* beginning *and the* end *of* τ*. The set of all time intervals over* $\mathfrak{T}$ *is denoted by* I*.*

Effectively, SQL:2011 provides direct support for the representation of what is traditionally known as concrete temporal databases, i.e., finite syntactic encodings of temporal data. The actual meaning of these encodings is captured by possibly infinite abstract temporal databases [8]. In the OBDA setting these two notions translate naturally into two types of ABoxes.

Definition 3 (Concrete and abstract temporal ABoxes). *A* temporal assertion *is an expression* $\tau : \alpha$*, where* α *is an ABox assertion and* $\tau \in I$*, stating that* α *is valid in every time point in* τ*. A* concrete temporal ABox *(CTA)* $\mathfrak{A}$ *is a finite set of temporal assertions. For a concrete temporal ABox* $\mathfrak{A}$ *its corresponding* abstract temporal ABox *(ATA) is a sequence* $(\mathfrak{A}_t)_{t \in T}$*, where* $\mathfrak{A}_t = \{\alpha \mid \tau : \alpha \in \mathfrak{A} \text{ and } t \in \tau\}$*.*

On the query level, SQL:2011 introduces a collection of predefined, binary, interval predicates to be used in the WHERE clauses of SQL queries for comparing pairs of time periods, e.g.: CONTAINS(τ_1, τ_2), OVERLAPS(τ_1, τ_2), PRECEDES(τ_1, τ_2), etc.

2.3 Ontology-based access to temporal data

We study ontology-based access to temporal data in the sense of an extension to the OBDA paradigm whose prototypical application could be illustrated with the following scenario.

Consider the SQL:2011 database presented in Table 1, with columns `from` and `to` marking the limits of the validity periods of the records. Such databases can be straightforwardly restated as temporal ABoxes, by applying the usual mapping from relational data to DLs [6], accompanied with an obvious mapping for handling time-stamps. Table 2 contains temporal ABoxes representing the data from the first record of the table `Emp`, for $T = \mathbb{N}$. We further envision a language for querying such temporal ABoxes, which combines support for two essential functionalities: representation of temporal constraints over the validity periods of data and semantically enhanced access to that data via ontologies. For instance, given the ontology

Emp				
id	name	department	from	to
e1	john	d1	1998	2000
e1	john	d3	2000	2003
e2	mark	d2	1999	2002

Dep				
id	type	location	from	to
d1	financial	madrid	1998	1999
d1	financial	barcelona	1999	2003
d2	hr	barcelona	2000	2003
d3	hq	london	2000	2003

Table 1: SQL:2011 temporal database.

$\mathfrak{A}$
$[1998, 2000] : \mathit{Emp}(\mathit{e1})$ $[1998, 2000] : \mathit{name}(\mathit{e1}, \mathit{john})$ $[1998, 2000] : \mathit{department}(\mathit{e1}, \mathit{d1})$

$(\mathfrak{A}_t)_{t<1998}$	$\mathfrak{A}_{1998}, \mathfrak{A}_{1999}, \mathfrak{A}_{2000}$	$(\mathfrak{A}_t)_{t>2000}$
$\emptyset$	$\mathit{Emp}(\mathit{e1})$ $\mathit{name}(\mathit{e1}, \mathit{john})$ $\mathit{department}(\mathit{e1}, \mathit{d1})$	$\emptyset$

Table 2: Temporal ABoxes: a CTA (left) and an ATA (right).

$\mathcal{T} = \{\mathit{Emp} \sqsubseteq \mathit{Person}, \mathit{department} \sqsubseteq \mathit{worksAt}, \mathit{location} \sqsubseteq \mathit{basedIn}\}$, the language should be able to support queries such as:

> (**Q**) *Find all persons x and periods y, such that x worked in a department based in Barcelona during y, but earlier x worked in a department based in Madrid.*

The expected answers should then include $\{x \mapsto \mathit{e1}, y \mapsto [1999, 2000]\}$. The practical rationale behind thus defined ontology-based access to temporal data is to eventually enable such queries to be translated to SQL:2011 and answered within existing RDBMSs.

3 Temporal Query Language

The language TQL, presented in this section, is defined using a generic construction method for temporal query languages in DLs, explored in [10, 3, 5]. TQL is a combination of a temporal language with CQs, obtained by substituting CQs for the atoms in temporal formulas. This design allows for very flexible interleaving of data queries with temporal constraints, while benefiting from the expressive power of both components. Here we use an interval-based variant of the temporal component, rather than a point-based, which facilitates direct querying of concrete databases, without requiring intermediate translations. The CQs are embedded in the temporal language using the epistemic semantics. This approach, suggested in [10], makes our proposal practical in the sense of immediately warranting the desirable rewritability properties of the resulting temporal queries.

As the first step, we compile out a part of the temporal component of TQL into custom interval predicates, interpreted here as predicates expressible in a simple algebra over time points. Note that on linear orders $u \leq v$ is definable as $\neg(v < u)$, while $v = u$ as $u \leq v \wedge v \leq u$.

Definition 4 (Interval predicates). *An n-ary* interval predicate *$\omega(\tau_1, \ldots, \tau_n)$ is given by its name ω and the defining condition expressed as a formula ϖ in the language:*

$$\varpi ::= u < v \mid \neg\varpi \mid \varpi_1 \wedge \varpi_2$$

where $u, v \in \{\tau_i^-, \tau_i^+ \mid 1 \leq i \leq n\}$. For a time domain $\mathfrak{T} = (T, <)$ and any $\tau_1, \ldots, \tau_n \in I$, a predicate $\omega(\tau_1, \ldots, \tau_n)$ evaluates to True *whenever $\varpi(\tau_1, \ldots, \tau_n)$ is satisfied in $\mathfrak{T}$ and to* False *otherwise, where $<$ is interpreted as the ordering in $\mathfrak{T}$ and $\neg, \wedge$ as usual. The set of all interval predicates is denoted by Ω.*

Next, we incorporate interval predicates in the temporal query language. By I_V we denote a countably infinite set of variables ranging over I.

Definition 5 (TQL)**.** Temporal query language *(TQL) is the smallest set of formulas constructed according to the grammar:*

$$\psi \ ::= [q](u) \ \mid \ \omega(u_1, \ldots, u_n) \ \mid \ \neg\psi \ \mid \ \psi_1 \wedge \psi_2 \ \mid \ \exists y.\psi$$

where $u, u_1, \ldots, u_n \in I \cup \mathsf{I}_\mathsf{V}$, $y \in \mathsf{I}_\mathsf{V}$, $q \in \mathcal{Q}_\Sigma$, and $\omega \in \Omega$. An I-substitution is a mapping $\pi : I \cup \mathsf{I}_\mathsf{V} \mapsto I$ such that $\pi(\tau) = \tau$ for every $\tau \in I$. By $\mathsf{obj}(\psi)$ we denote the set of free individual variables and by $\mathsf{int}(\psi)$ the set of free interval variables in ψ. A TQL formula ψ is grounded *iff $\mathsf{obj}(\psi) = \mathsf{int}(\psi) = \emptyset$. The satisfaction relation for grounded TQL formulas, w.r.t. a TBox $\mathcal{T}$, a CTA $\mathfrak{A}$, and an I-substitution π, is defined inductively as follows:*

$$\begin{array}{lll}
(\dagger) \ \mathcal{T}, \mathfrak{A}, \pi \models [q](u) & \text{iff} & \mathcal{T}, \mathfrak{A}_t \models q, \textit{ for every } t \in \pi(u), \\
\mathcal{T}, \mathfrak{A}, \pi \models \omega(u_1, \ldots, u_n) & \text{iff} & \varpi(\pi(u_1), \ldots, \pi(u_n)), \\
\mathcal{T}, \mathfrak{A}, \pi \models \neg\psi & \text{iff} & \mathcal{T}, \mathfrak{A}, \pi \not\models \psi, \\
\mathcal{T}, \mathfrak{A}, \pi \models \psi_1 \wedge \psi_2 & \text{iff} & \mathcal{T}, \mathfrak{A}, \pi \models \psi_1 \textit{ and } \mathcal{T}, \mathfrak{A}, \pi \models \psi_2, \\
\mathcal{T}, \mathfrak{A}, \pi \models \exists y.\psi & \text{iff} & \textit{there exists } \tau \in I, \textit{ such that} \\
 & & \mathcal{T}, \mathfrak{A}, \pi[y \mapsto \tau] \models \psi,
\end{array}$$

where $\pi[y \mapsto \tau]$ denotes the I-substitution exactly as π except for that we fix $\pi(y) = \tau$. We say that $\mathcal{T}, \mathfrak{A}$ entail *a grounded TQL formula ψ, denoted as $\mathcal{T}, \mathfrak{A} \models \psi$,* iff *there exists an I-substitution π, such that $\mathcal{T}, \mathfrak{A}, \pi \models \psi$.*

TQL formulas with free variables can naturally serve as queries over concrete temporal ABoxes. We refer to such formulas as *concrete TQL queries* (CTQs). As an example of a CTQ, consider a rephrasing of the query (**Q**) from Section 2.3:

$$\begin{array}{l}\psi(x, y) := [\exists z.(Person(x) \wedge worksAt(x, z) \wedge basedIn(z, barcelona))](y) \wedge \\ \exists v.(\text{PRECEDES}(v, y) \wedge [\exists z.(worksAt(x, z) \wedge basedIn(z, madrid))](v))\end{array}$$

where the meaning of PRECEDES is as expected. As one of its answers, $\psi(x, y)$ should return $\{x \mapsto \mathit{e1}, y \mapsto [1999, 2000]\}$, which is a compact representation of all answers to this query over the temporal database in Section 2.3.

Finally, we define the certain answer semantics for TQL queries.

Definition 6 (CTQ answering)**.** *Let $\mathcal{T}$ be a TBox, $\mathfrak{A}$ a CTA and ψ a CTQ with free variables $\mathsf{obj}(\psi)$ and $\mathsf{int}(\psi)$. An* answer *to ψ is a mapping σ such that $\sigma : \mathsf{obj}(\psi) \mapsto \mathsf{N_I}$ and $\sigma : \mathsf{int}(\psi) \mapsto I$. By $\sigma(\psi)$ we denote the result of uniformly substituting every occurrence of x in ψ with $\sigma(x)$, for every $x \in \mathsf{obj}(\psi) \cup \mathsf{int}(\psi)$. An answer σ is called* certain *over $\mathcal{T}$, $\mathfrak{A}$* iff *$\mathcal{T}, \mathfrak{A} \models \sigma(\psi)$.*

4 Query rewriting

The key to the formal design of TQL is condition ($\dagger$), in Definition 5, which ensures the epistemic interpretation of the CQs embedded in TQL queries. The formula $[q](\tau)$, for a grounded $q \in \mathcal{Q}_\Sigma$ and $\tau \in I$, reads as "*q is entailed in all time points in τ*". Analogically, $\neg[q]$ is interpreted as negation-as-failure: "*it is not true that q is entailed in all time points in τ*". As a consequence, condition (†) can be effectively replaced with its equivalent, which involves the standard FO rewritability techniques in the sense of Definition 1:

$$\mathcal{T}, \mathfrak{A}, \pi \models [q](u) \quad \text{iff} \quad db(\mathfrak{A}_t) \Vdash q^{\mathcal{T}}, \text{ for every } t \in \pi(u),$$

where $q^{\mathcal{T}}$ is an FO rewriting of q in $\mathcal{T}$. What it eventually means, is that all occurrences of CQs in a CTQ can be replaced with their FO rewritings, so that the ontology $\mathcal{T}$ can be dropped, while the resulting formula can be evaluated exclusively over the temporal ABox seen as a sequence of FO interpretations. In what follows, we make this observation formally precise. Essentially, we rewrite CTQs into a special temporal FO form, and show that such rewritings preserve the correctness of query answering. Importantly, the FO component in the rewritings is defined purely in terms of the standard FO rewritings of the respective CQs. To ease the proof of the subsequent complexity result, the temporal component is moreover rephrased from an interval- to point-based.

Definition 7 (TFO rewriting of CTQs)**.** *A* temporal FO rewriting *of a CTQ ψ is a formula $\lceil\psi\rceil^{TFO}$ obtained from ψ by applying the transformation $\lceil\cdot\rceil^{TFO}$, defined inductively as follows:*

$$\begin{array}{rcl}
\lceil [q](u) \rceil^{TFO} & = & \forall x.(u^- \leq x \leq u^+ \rightarrow (q^{\mathcal{T}})(x)),\\
\lceil \omega(\vec{u}) \rceil^{TFO} & = & \varpi(\vec{u}),\\
\lceil \neg\psi \rceil^{TFO} & = & \neg\lceil\psi\rceil^{TFO},\\
\lceil \psi_1 \wedge \psi_2 \rceil^{TFO} & = & \lceil\psi_1\rceil^{TFO} \wedge \lceil\psi_2\rceil^{TFO},\\
\lceil \exists y.\psi \rceil^{TFO} & = & \exists y^-, y^+.(y^- \leq y^+ \wedge \lceil\psi\rceil^{TFO}).
\end{array}$$

By $\mathsf{obj}(\lceil\psi\rceil^{TFO})$ we denote the set of free individual variables and by $\mathsf{pt}(\lceil\psi\rceil^{TFO})$ the set of free time point variables in $\lceil\psi\rceil^{TFO}$. For every answer σ to ψ, we define the corresponding answer σ^* to $\lceil\psi\rceil^{TFO}$, such that $\sigma^* : \mathsf{obj}(\lceil\psi\rceil^{TFO}) \mapsto \mathsf{N_I}$ and $\sigma^* : \mathsf{pt}(\lceil\psi\rceil^{TFO}) \mapsto T$, where:

- *$\sigma^*(x) := \sigma(x)$, for every $x \in \mathsf{obj}(\psi)$,*
- *$\sigma^*(x^-) := \sigma(x)^-$ and $\sigma^*(x^+) := \sigma(x)^+$, for every $x \in \mathsf{int}(\psi)$.*

As usually, we assume that $\forall x.\phi = \neg\exists x.\neg\phi$ and $\phi_1 \rightarrow \phi_2 = \neg(\phi_1 \wedge \neg\phi_2)$. Note that whenever $x \in \mathsf{I_V}$ then $x^-, x^+ \in \mathsf{T_V}$ are two variables uniquely associated with x. Consequently, given the transformation rules, there exists a one-to-one correspondence between answers to CTQs and to their TFO rewritings.

The semantics of the rewriting $\lceil\psi\rceil^{TFO}$ is defined over a sequence of databses/FO interpretations $(db(\mathfrak{A}_t))_{t\in T}$, based on the CTA $\mathfrak{A}$, where every $db(\mathfrak{A}_t)$ is specified as in Definition 1.

Definition 8 (TFO semantics)**.** *A T-substitution is a mapping $\pi : T \cup \mathsf{T_V} \mapsto T$ such that $\pi(t) = t$ for every $t \in T$. Whenever $\mathsf{obj}(\lceil\psi\rceil^{TFO}) = \mathsf{pt}(\lceil\psi\rceil^{TFO}) = \emptyset$ the satisfaction relation for $\lceil\psi\rceil^{TFO}$, w.r.t. a T-substitution π, is defined inductively as follows:*

$$\begin{array}{lll}
(db(\mathfrak{A}_t))_{t\in T}, \pi \models (q^{\mathcal{T}})(u) & \text{iff} & db(\mathfrak{A}_{\pi(u)}) \Vdash q^{\mathcal{T}},\\
(db(\mathfrak{A}_t))_{t\in T}, \pi \models u < v & \text{iff} & \pi(u) < \pi(v),\\
(db(\mathfrak{A}_t))_{t\in T}, \pi \models \neg\phi & \text{iff} & (db(\mathfrak{A}_t))_{t\in T}, \pi \not\models \phi,\\
(db(\mathfrak{A}_t))_{t\in T}, \pi \models \phi_1 \wedge \phi_2 & \text{iff} & (db(\mathfrak{A}_t))_{t\in T}, \pi \models \phi_1 \textit{ and}\\
 & & (db(\mathfrak{A}_t))_{t\in T}, \pi \models \phi_2,\\
(db(\mathfrak{A}_t))_{t\in T}, \pi \models \exists y.\phi & \text{iff} & \textit{there exists } t \in T, \textit{ such that}\\
 & & (db(\mathfrak{A}_t))_{t\in T}, \pi[y \mapsto t] \models \phi,
\end{array}$$

where $\pi[y \mapsto t]$ denotes the T-substitution exactly as π except for that we fix $\pi(y) = t$. We say that a grounded TFO formula $\lceil\psi\rceil^{TFO}$ is satisfied *in $(db(\mathfrak{A}_t))_{t\in T}$, denoted as $(db(\mathfrak{A}_t))_{t\in T} \models \lceil\psi\rceil^{TFO}$,* iff *there exists a T-substitution π, such that $(db(\mathfrak{A}_t))_{t\in T}, \pi \models \lceil\psi\rceil^{TFO}$.*

The following theorem ensures the correctness of query answering under the TFO rewriting. The proof, included in Appendix, follows by structural induction over the syntax of CTQs and their rewritings.

Theorem 1 (Correctness of TFO rewriting)**.** *Let ψ be a CTQ and $\mathcal{T}$ a TBox. Then for every answer σ to ψ and CTA $\mathfrak{A}$ it holds that:*

$$\mathcal{T}, \mathfrak{A} \models \sigma(\psi) \;\text{ iff }\; (db(\mathfrak{A}_t))_{t\in T} \models \sigma^*(\lceil\psi\rceil^{TFO}).$$

Given some further shallow refinements to the resulting TFO form, TQL can be easily seen as a fragment of two-sorted FOL, or alternatively, as a fragment of First-Order Temporal Logic [12]. Both types of languages have been studied in the context of temporal databases [7]. The data complexity of query answering in such formalisms depends strongly on the choice of the underlying time domain and can range from AC^0 to undecidable. A natural strategy to ensure the desirable AC^0 complexity is to restrict time domains to finite ones only. This result, presented in Theorem 2, follows by the correspondence of TQL to yet another known formalism: FOL with restricted arithmetic [13].

Theorem 2 (Complexity)**.** *The data complexity of answering CTQs in the presence of TBoxes, and over finite time domains $\mathfrak{T} = (T, <)$ is in* AC^0.

Under the finite time domain assumption, the resulting TFO queries can be effectively translated into SQL:2011 and evaluated using existing RDBMSs with SQL:2011 support. This final translation, and its efficient generation by means of existing CQ rewriting tools, is the ultimate goal of the studied scenario, and remains part of our future research agenda.

5 Related work

Part of work related to this research comes from the field of temporal databases [7, 8]. Although providing some theoretical grounding for our proposal, these contributions are obviously agnostic about the ontology-based approach on the problem, which we concentrate on here.

The research on temporal extensions to OBDA has been taken up only very recently by Borgwardt et al. [3, 5] and Artale et al. [2]. In [3], the authors study the same prototypical scenario as addressed here, but focus on its more foundational aspects. They consider a more expressive DL $\mathcal{ALC}$ as the background ontology language and adopt a less restrictive definition of temporal queries. This offers a richer setting, yet without apparent application prospects in the context of existing temporal databases. The work in [5] is closer aligned with ours. It considers DL-*Lite* ontologies and studies a rewriting approach based on the use of standard CQ rewriting techniques. It remains complementary to the present contribution in that it employs a somewhat different query language (no epistemic interpretation of embedded CQs, no negation), and does not focus on the task of querying temporal data via SQL:2011, which is the guiding motive for our work. In [2], the authors pursue an alternative approach to temporal OBDA. Instead of adding temporal features to query languages, they focus on defining suitable temporal extensions to OWL 2 QL, in the spirit of known temporal DLs [1]. The practical implications of this work have not been yet investigated.

6 Conclusions

We propose a practical approach to lifting the popular paradigm of OBDA to temporal case. The presented language TQL allows for querying temporal data stored in relational databases via a semantic layer of OWL 2 QL ontologies. We believe that this proposal, to be further developed in future research, strikes a good balance between the theoretical strength of its formal foundations and the feasibility of practical applications, warranted by the possibility of answering TQL queries in commercially supported RDBMSs via a translation to SQL:2011.

From a more general perspective, considering the growing interest in temporal extensions of OBDA, it is critical to continue the formal study of temporal features supported by the SQL standards, temporal extensions to logic-based query languages and ontology languages intended for use in practical OBDA scenarios, and finally, the relationships holding between all of them. In this respect, it is highly desirable to establish stronger links between the approach pursued here and those proposed recently in [3, 5, 2], with the prospect of gaining a clearer view on the landscape of technical possibilities regarding the temporal OBDA.

References

[1] Alessandro Artale, Enrico Franconi, Frank Wolter, and Michael Zakharyaschev. A temporal description logic for reasoning over conceptual schemas and queries. In *Proc. of the European Conference on Logics in Artificial Intelligence (JELIA-02)*, 2002.

[2] Alessandro Artale, Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. Temporal description logic for ontology-based data access. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI-13)*, 2013.

[3] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. Temporalizing ontology-based data access. In *Proceedings of the International Conference on Automated Deduction (CADE-13)*, 2013.

[4] Franz Baader, Diego Calvanese, Deborah L. Mcguinness, Daniele Nardi, and Peter F. Patel-Schneider. *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, 2003.

[5] Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. Temporal query answering in the description logic DL-Lite. In *Proc. of the International Symposium on Frontiers of Combining Systems (FroCoS-13)*, 2013.

[6] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.

[7] Jan Chomicki. Temporal query languages: A survey. In *Proc. of the International Conference on Temporal Logic (ICTL-94)*, 1994.

[8] Jan Chomicki and David Toman. Temporal Databases. In *Handbook of Temporal Reasoning in Artificial Intelligence (Foundations of Artificial Intelligence)*, pages 429–468. Elsevier Science Inc., 2005.

[9] Birte Glimm, Ian Horrocks, Carsten Lutz, and Uli Sattler. Conjunctive query answering for the description logic SHIQ. In *Journal of Artificial Intelligence Research*, 2008.

[10] Víctor Gutiérrez-Basulto and Szymon Klarman. Towards a unifying approach to representing and querying temporal data in description logics. In *Proc. of the International Conference on Web Reasoning and Rule Systems (RR-12)*, 2012.

[11] Krishna Kulkarni and Jan-Eike Michels. Temporal features in SQL:2011. *SIGMOD Rec.*, 41(3), 2012.

[12] Mark Reynolds. The complexity of decision problems for linear temporal logics. *Journal of Studies in Logic*, 3(1), 2010.

[13] Nicole Schweikardt. Arithmetic, first-order logic, and counting quantifiers. *ACM Transactions on Computational Logic*, 6(3), 2005.

Appendix

Proof of Theorem 1. As remarked under Definition 7, there exists a one-to-one correspondence between answers to CTQs and to their TFO rewritings, simply because every free interval variable is replaced with a pair of time point variables, binding over the beginning and the end of the interval. We can further identify a similar correspondence between I- and T-substitutions for the respective formulas. For an I-substitution π we define T-substitution π^*, by fixing:

$$\pi^*(u^-) := \pi(u)^- \text{ and } \pi^*(u^+) := \pi(u)^+, \text{ for every } u \in I \cup \mathsf{I_V}.$$

Note, however, that TFO rewritings might contain extra variables added in the transformation of expressions $[q](u)$ into $\forall x.(u^- \leq x \leq u^+ \to (q^{\mathcal{T}})(x))$. Consequently, an I-substitution π can be uniquely associated with a set of T-substitutions π^*, differing with the assignment of every such variable x, which is set as $\pi^*(x) := t$, for some $\pi^*(u^-) \leq t \leq \pi^*(u^+)$.

Based on these correspondences, we show that for every pair of answers σ and σ^* to a CTQ and its TFO rewriting, the former formula is satisfied *iff* the latter is, under the respective semantics, w.r.t. corresponding substitutions.

($\Rightarrow$) Let ψ be a CTQ and $\mathcal{T}, \mathfrak{A} \models \sigma(\psi)$, for some $\sigma, \mathcal{T}$ and $\mathfrak{A}$. We show that $(db(\mathfrak{A}_t))_{t\in T} \models \sigma^*(\lceil\psi\rceil^{TFO})$. For simplicity, we assume that ψ and $\lceil\psi\rceil^{TFO}$ are already grounded with the respective answers. By Definition 6, there must exist an I-substitution π, such that $\mathcal{T}, \mathfrak{A}, \pi \models \psi$. We proceed by structural induction to show that $(db(\mathfrak{A}_t))_{t\in T}, \pi^* \models \lceil\psi\rceil^{TFO}$. The first case is the critical one. Suppose $\mathcal{T}, \mathfrak{A}, \pi \models [q](u)$, and therefore, that $\mathcal{T}, \mathfrak{A}_t \models q$, for every $t \in \pi(u)$. Then it should hold that $(db(\mathfrak{A}_t))_{t\in T}, \pi^* \models \forall x.(u^- \leq x \leq u^+ \to (q^{\mathcal{T}})(x))$. But this follows immediately by the semantics of $\lceil\psi\rceil^{TFO}$, definition of π^*, and the FO rewritability of CQs (Definition 1). Observe that for $\pi^*(x) = t \in \pi(u)$ it is indeed the case that $(db(\mathfrak{A}_t))_{t\in T}, \pi^* \models u^- \leq x \leq u^+$, and so that $db(\mathfrak{A}_{\pi^*(x)}) \Vdash q^{\mathcal{T}}$, as clearly $\mathcal{T}, \mathfrak{A}_t \models q$. For $\mathcal{T}, \mathfrak{A}, \pi \models \omega(u_1, \dots, u_n)$, the satisfaction of $\varpi(\vec{u})$ is immediate by the semantics of TQL. The cases of $\neg, \wedge$ are straightforward. For $\exists y.\psi$, observe that whenever there exists $\tau \in I$, such that $\mathcal{T}, \mathfrak{A}, \pi[y \mapsto \tau] \models \psi$, then there must also exist $\tau^-, \tau^+ \in T$, such that $\tau^- \leq \tau^+$ and $(db(\mathfrak{A}_t))_{t\in T}, \pi[y_1 \mapsto \tau^-, y_2 \mapsto \tau^+] \models \phi$.

($\Leftarrow$) This direction follows by the inverse argument to the above, and rests on the same correspondences between T- and I-substitutions. ❑

Proof of Theorem 2. We observe that the language of TFO rewritings of CTQs can be also seen as FOL with restricted arithmetic, where T represents a set of numbers and $<$ the less-than predicate. To emphasize that view we slightly revise the syntax of the CTAs and the TFO rewritings, and internalize the temporal argument into all predicates. We define $\mathfrak{A}^{\mathbb{N}} = \bigcup_{t\in T} \mathfrak{A}_t^{\mathbb{N}}$, where $\mathfrak{A}_t^{\mathbb{N}} = \{A(a,t) \mid A(a) \in \mathfrak{A}_t\} \cup \{r(a,b,t) \mid r(a,b) \in \mathfrak{A}_t\}$. Then by $db(\mathfrak{A}^{\mathbb{N}})$ we denote $\mathfrak{A}^{\mathbb{N}}$ considered as a database/FO interpretation, analogically as specified in Definition 1. Further, we write $q^{\mathcal{T}}(x)^{\mathbb{N}} = \bigvee_i q_i(x)^{\mathbb{N}}$, where $q^{\mathcal{T}} = \bigvee_i q_i$ and $q_i(x)^{\mathbb{N}}$ is obtained from q_i, by replacing every atom $A(u)$ and $r(u,v)$ in q_i with $A(u,x)$ and $r(u,v,x)$ respectively. The first TFO transformation rule is revised to:

$$\lceil [q](u) \rceil^{TFO} = \forall x.(u^- \leq x \leq u^+ \to (q^{\mathcal{T}}(x)^{\mathbb{N}}),$$

The FO rewriting is interpreted over $\mathfrak{A}^{\mathbb{N}}$ via the standard FO satisfaction relation. In particular $q^{\mathcal{T}}(x)^{\mathbb{N}}$ is interpreted as $\mathfrak{A}^{\mathbb{N}} \Vdash q^{\mathcal{T}}(x)^{\mathbb{N}}$.

The data complexity of FOL with arithmetic $(\underline{N}, <)$, where $\underline{N} \subseteq \mathbb{N}$ is a finite initial segment of natural numbers $\mathbb{N}$, is known to be AC^0 [13]. Clearly, every finite time domain $\mathfrak{T} = (T, <)$ can be isomorphically mapped to the initial segment $\{0, 1, \dots, |T| - 1\}$ of $\mathbb{N}$. Therefore, query answering in TQL over such domains corresponds to query answering in FOL with $(\underline{N}, <)$, and must belong, as a consequence, to the same complexity class. ❑