Towards a Unifying Approach to Representing and Querying Temporal Data in Description Logics

Víctor Gutiérrez-Basulto[1] and Szymon Klarman[2]

[1] Department of Computer Science, Universität Bremen
victor@informatik.uni-bremen.de
[2] Department of Computer Science, Vrije Universiteit Amsterdam
s.klarman@vu.nl

Abstract. Establishing a generic approach to representing and querying temporal data in the context of Description Logics (DLs) is an important, and still open challenge. The difficulty lies in that a proposed approach should reconcile a number of valuable contributions coming from diverse, yet relevant research lines, such as temporal databases and query answering in DLs, but also temporal DLs and Semantic Web practices involving rich temporal vocabularies. Within such a variety of influences, it is critical to carefully balance theoretical foundations with good prospects for reusing existing techniques, tools and methodologies. In this paper, we attempt to make first steps towards this goal. After providing a comprehensive overview of the background research and identifying the core requirements, we propose a general mechanism of defining temporal query languages for time-stamped data in DLs, based on combinations of linear temporal logics with first-order queries. Further, we advocate a controlled use of epistemic semantics in order to warrant practical query answering. We systematically motivate our proposal and highlight its basic theoretical and practical implications. Finally, we outline open problems and key directions for future research.

1 Introduction

The use of Description Logic (DL) ontologies for describing and interpreting data is acknowledged by now as a self-standing paradigm of data management in different areas of computer science — most prominently on the Semantic Web (SW), where DL-based ontology languages play a key architectural role. One big and yet unresolved challenge in this context, called for by numerous applications, is to formally incorporate and operationalize the notion of data's *validity time*, i.e. the explicitly declared time span within which the data is known to be true.

Problem: In this paper, we study the problem of managing temporal data in the framework of DLs. Our goal is to make first steps towards establishing a unifying approach to representing and querying such data under DL ontologies. Given the multifaceted nature of the problem and the scope of expected applications, one

of main challenges which must be faced lies in reconciling a number of valuable contributions developed within diverse research areas. In particular:

- *temporal databases*: for ensuring commensurability with the commonly adopted temporal data models for representing validity time and with standard query languages based on temporal first-order logic,
- *query answering* in DLs: for enabling transfer of known query answering techniques, complexity results, and facilitating reuse of existing tools,
- *temporal DLs*: for enabling the possibility of managing temporal data under DL ontologies which capture temporal constraints on the intensional level,
- SW *temporal vocabularies*: for supporting typical SW practices involving OWL-based time ontologies, which provide rich temporal vocabularies employed on the level of queries and data annotations.

Clearly, under such a variety of influences, it is critical to carefully balance theoretical foundations of a proposed approach with good prospects for reusing existing techniques, tools and methodologies.

Contributions: We introduce a basic framework for representing temporal data in arbitrary DLs, where the data takes the form of time-stamped ABox assertions $[t_1, t_2] : \alpha$, stating validity of the assertion α during the interval $[t_1, t_2]$. Then we propose a general mechanism of defining corresponding temporal query languages, based on combinations of linear temporal logics with classes of first-order queries — specifically, with well-known conjunctive queries. In particular:

- we systematically motivate the proposed mechanism, present the syntax and certain answer semantics for the query languages that the mechanism generates, and the relationship of those languages to temporal first-order logic.
- we advocate a controlled use of epistemic semantics in order to warrant practical query answering in the defined setting. Under this restriction, we deliver a $\text{PSPACE}^{QA(\mathcal{L})}$-completeness bound for the combined complexity of answering temporal queries in an arbitrary DL $\mathcal{L}$, where $QA(\mathcal{L})$ is an oracle answering conjunctive queries in $\mathcal{L}$. We highlight some essential theoretical and practical implications of this result.
- we discuss the possibility of pushing the approach further towards integration with temporal DLs and SW temporal vocabularies.

Structure of the paper: In Section 2, we provide a comprehensive overview of the background research and identify the core requirements for the proposed approach. Next, we discuss DL preliminaries in Section 3 and introduce the temporal data model in Section 4. In Section 5, we present and study the proposed mechanism of defining temporal query languages. In Section 6 we discuss similarities to existing approaches and outline some future research directions. We conclude the paper in Section 7.

2 Overview and Background

Extending information systems with capabilities for managing temporal information has been deeply studied and advocated in many areas of computer science, particularly, in those concerned with relational databases and knowledge representation. Surprisingly, despite the successful use of the ontology-based data access (OBDA) paradigm as an application of DL technologies in databases, the development of mechanisms for extending the OBDA approach towards accessing temporal data have not been yet investigated. A proposed mechanism should naturally take into account the already well-founded research lines on representing and querying temporal information, as well as valuable contributions in related areas, which we outline in the following paragraphs.

Ontology-Based Data Access The *ontology-based data access* is a paradigm of managing data in presence of background knowledge, represented as a formal ontology, enabling convenient query answering over *incomplete* data. In recent years, special attention has been given to ontologies based on DL languages. A considerable amount of research has been devoted to the problem of query answering in DLs, focusing predominantly on *conjunctive queries* (CQs). This has lead to establishing a clear picture of the computational complexity of CQ answering, and to the development of algorithmic approaches. The study has been focused on two major lines: 1) utilization of classical DLs with high expressive power, where the complexity of query answering turns out typically too high for practical applications [1]; 2) development of DLs allowing efficient query answering over large amounts of data. Calvanese et.al. [2] introduced the DL-Lite family of DLs, for which efficient OBDA can be achieved by reduction to query answering in relational database management systems (RDBMSs). One of the key motivations behind the design of the temporal query languages presented in this paper is to enable easy, modular reuse of the known techniques and results on query answering in DLs in the context of temporal data querying.

Temporal Databases During the 90s, the database community conducted an exhaustive study on temporal extensions of the standard relational data models, supporting management of temporal information. The common way of constructing *temporal relational databases* (TDBs) is to enrich traditional data models with *time-stamps* representing data's validity time, i.e. the time span within which the data is known to be true. As one of the crucial requirements for our approach we pose formal compatibility with the TDB paradigm of representing temporal data. Inspired by the notion of *concrete temporal database* [3], we construct a *temporal ABox* by time-stamping every ABox assertion with a weak-interval of the form $[t_1, t_2]$, compactly representing a set of time points in which the assertion is valid. The semantics of a temporal ABox, by analogy to TDBs case [3], is given by mapping each time point in the underlying *time domain* to the non-temporal (standard) ABox — a so-called *snapshot* — containing exactly the assertions valid in that point. Eventually, the OBDA paradigm is applied within the scope of respective snapshots.

Regarding the choice of the time domain, the TDB literature reports on a number of possible representations, each one having far-reaching philosophical, logical and computational consequences [4]. The available degrees of freedom concern, among others: the nature of the atomic time entities (points *vs.* intervals), their ordering relationships (linear *vs.* branching *vs.* partial orders), the density (discrete *vs.* continuous), the boundaries (finite *vs.* infinite). Although strict commitment to any representation is always arbitrary to some extent, arguably one of the most natural and commonly used setups in TDBs, which we also adopt here, is the one capturing the intuition of a point-based time line [4].

A temporal data model is complemented by an adequate *temporal query language* for querying temporal data. In this aspect, we ground our proposal in two well-known research lines. 1) Following the research on TDBs, we consider languages based on fragments of *temporal first-order logic*, which has been advocated as a suitable high-level formalism for querying TDBs [3]. It has been shown, that queries expressed in temporal first-order logic can be translated directly to TSQL2 [5] — a temporal extension of the standard database query language SQL — and thus efficiently handled using existing TDB systems. 2) Given the known landscape of complexity results and developed techniques for query answering in DLs, we pay special attention to the expressiveness of the first-order component within the intended fragments of temporal first-order logic. As explained in detail in Section 5, our motivation is to provide a mechanism for defining such fragments in a controlled, modular manner, by selecting particular sets of temporal operators and particular classes of first-order queries to be combined. By specifying those two parameters one should effectively obtain a ready query language of a well-characterized computational behavior. To this end we make use of the methodology of temporalizing logic systems [6].

Semantic Web & Temporal DLs In recent years, the problem of managing time-varying knowledge has gained a lot of interest also in the Semantic Web research community. Particularly, the need for describing temporal information on the Web gave rise to various *time ontologies* [7], which formalize common temporal notions, such as temporal instants, temporal intervals and calendar terms, and offer standardized formats for representing different types of temporal information. Although such ontologies succeed in facilitating exchange of time-oriented data among Web agents, they are not accompanied by any formally grounded methodologies of processing such information. Specifically, they offer no inference mechanisms to support genuinely temporal reasoning. This lack of rigorous logical foundations, is in practical scenarios partially remedied by the use of programming tools and ad-hoc hybrid architectures [8,9].

Some alternative approaches, building more systematically on the TDB philosophy, were also proposed for representing and querying temporal data in RDFs [10,11] and OWL [12]. Although employing the same or similar temporal data models as in our case, these frameworks are mostly technology-driven and do not consider the design of query languages in sufficient generality.

A somewhat orthogonal research effort has gone into designing a family of *temporal description logics* (TDLs) [13] tailored for representing and reasoning

with inherently temporal terminologies. As proper combinations of temporal logics with DLs, TDLs count with a well-defined temporal semantics, which makes them very appealing from the theoretical perspective. Nevertheless, most of the contributions in this area focus on traditional reasoning tasks such as satisfiability and subsumption, related mostly to conceptual modeling rather than querying temporal data, with very few, limited exceptions [14]. In general, a potential transfer of the known query answering techniques for DLs to the TDL setting seems highly non-trivial.

Although in this paper we do not address the problems related to querying temporal data with support of SW temporal vocabularies or in presence of TDL ontologies, we do acknowledge them as worthwhile challenges for future research, and we briefly reconsider them in Section 6.

3 Description Logic Preliminaries

We use the standard nomenclature and notation for the syntax and semantics of DLs (see [15] for full details). A DL language $\mathcal{L}$ is defined over a vocabulary $\Sigma = (\mathsf{N_C}, \mathsf{N_R}, \mathsf{N_I})$, where $\mathsf{N_C}, \mathsf{N_R}, \mathsf{N_I}$ are countably infinite sets of concept names, role names and individual names, respectively. By convention, we use letters A, B to denote concept names, r, s for role names and a, b for individual names. The grammar for complex concepts, roles and axioms is defined relative to a given DL dialect. For instance, the DL $\mathcal{ALC}$ provides the following concept constructors:

$$\neg C \;\mid\; C \sqcap D \;\mid\; C \sqcup D \;\mid\; \exists r.C \;\mid\; \forall r.C$$

where C, D are (possibly complex) concepts. A TBox $\mathcal{T}$ is a finite set of concept inclusions of the form $C \sqsubseteq D$, whereas an ABox $\mathcal{A}$ is a finite set of assertions of types $A(a)$ and $r(a, b)$. By following the DL-based OBDA paradigm, we consider a TBox to be the ontology through which one accesses the data represented as an ABox.

The semantics of $\mathcal{L}$ is given through interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty domain of individuals and $\cdot^{\mathcal{I}}$ is an interpretation function, which maps $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, for every $A \in \mathsf{N_C}$, $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, for every $r \in \mathsf{N_R}$, and $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, for every $a \in \mathsf{N_I}$, and is inductively extended over complex expressions according to the fixed conditions associated with each constructor [15]. An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ (resp. ABox $\mathcal{A}$), written $\mathcal{I} \models \mathcal{T}$ (resp. $\mathcal{I} \models \mathcal{A}$) *iff* it satisfies all the concept inclusions in $\mathcal{T}$ (resp. assertions in $\mathcal{A}$), where the satisfaction relation for concept inclusions and assertions is defined in the usual way. An ABox $\mathcal{A}$ is consistent w.r.t. a TBox $\mathcal{T}$ *iff* there exists an interpretation $\mathcal{I}$ which is a model of both $\mathcal{A}$ and $\mathcal{T}$, written as $\mathcal{I} \models \mathcal{T}, \mathcal{A}$.

Next, we recall the standard notion of conjunctive queries — the most commonly studied query formalism in the context of DLs [1]. 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, *answer 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.

Let $\mathcal{I}$ be an interpretation and $q(\vec{x})$ a CQ with the answer variables $\vec{x} = x_1, \ldots, x_k$. By term$(q)$ we denote the set of all terms occurring in q. For a sequence $\vec{a} = a_1, \ldots, a_k \in \mathsf{N_I}$, an $\vec{a}$-match to a query $q(\vec{x})$ in $\mathcal{I}$ is a mapping $\mu : \text{term}(q) \mapsto \Delta^{\mathcal{I}}$, such that $\mu(x_i) = {a_i}^{\mathcal{I}}$, for every $1 \leq i \leq k$, $\mu(a) = a^{\mathcal{I}}$, for every $a \in \mathsf{N_I} \cap \text{term}(\mathsf{q})$, for every $A(u)$ in q it is the case that $\mu(u) \in A^{\mathcal{I}}$ and for every $r(u, v)$ in q it is the case that $(\mu(u), \mu(v)) \in r^{\mathcal{I}}$. We write $\mathcal{I} \models q[\vec{a}]$ whenever there exists an $\vec{a}$-match to q in $\mathcal{I}$, and $\mathcal{T}, \mathcal{A} \models q[\vec{a}]$ whenever there exists an $\vec{a}$-match to q in every model of $\mathcal{T}$ and $\mathcal{A}$. In the latter case $\vec{a}$ is called a *certain answer* to q *w.r.t.* $\mathcal{T}$ and $\mathcal{A}$.

In what follows we implicitly assume that all considered TBoxes, ABoxes and CQs are expressed over the same DL vocabulary Σ.

4 Temporal Data Model

A temporal data model is formally specified by two basic characteristics: the choice of the underlying time domain and the syntax and semantics of temporal annotations linking data to a time domain. As outlined in Section 2, a time domain permitted in our scenario is a structure defined as a linear ordering of a set of time instants [4].

Definition 1 (Time domain). *A* time domain *is a tuple $(T, <)$, where T is a nonempty set of elements called* time instants *and $<$ is an irreflexive, linear ordering on T.*

Some popularly considered structures satisfying this definition are based on sets of numbers, e.g. naturals $(\mathbb{N}, <)$, integers $(\mathbb{Z}, <)$, reals $(\mathbb{R}, <)$, with the ordering $<$ being interpreted as the usual *smaller-than* relation. By convention, we write $\leq$ to denote the reflexive closure of $<$.

Temporal annotations are based on the weak-interval time-stamping mechanism. Intuitively, a time-stamped ABox assertion $[t_1, t_2] : \alpha$ states that the axiom α is valid in all time instants within the interval $[t_1, t_2]$. Additionally, we allow special symbols $-\infty$ and $+\infty$ to represent possibly unbounded intervals.

Definition 2 (Temporal ABox). *Let $(T, <)$ be a time domain. A* temporal assertion *is an expression in one of the following forms:*

$$[t_1, t_2] : \alpha \;\mid\; [-\infty, t_1] : \alpha \;\mid\; [t_1, +\infty] : \alpha \;\mid\; [-\infty, +\infty] : \alpha$$

where α is an ABox assertion and $t_1, t_2 \in T$. A temporal ABox *$\mathcal{A}_T$ is a finite set of temporal ABox assertions. A t-*snapshot *of $\mathcal{A}_T$, for $t \in T$, is the smallest ABox $\mathcal{A}_T(t)$ containing all assertions α, for which any of the following conditions hold:*

$$\begin{aligned} [t_1, t_2] : \alpha \in \mathcal{A} \quad &\textit{and} \quad t_1 \leq t \leq t_2, \\ [-\infty, t_1] : \alpha \in \mathcal{A} \quad &\textit{and} \quad t \leq t_1, \\ [t_1, +\infty] : \alpha \in \mathcal{A} \quad &\textit{and} \quad t_1 \leq t, \\ [-\infty, +\infty] : \alpha \in \mathcal{A}. & \end{aligned}$$

The standard DL semantics is extended in a natural way by adding the temporal dimension and assigning a single DL interpretation to every time instant.

Definition 3 (Snapshot semantics). *Let $(T, <)$ be a time domain, $\mathcal{T}$ a TBox and $\mathcal{A}_T$ a temporal ABox. A* temporal interpretation *of $\mathcal{T}$ and $\mathcal{A}_T$ is a tuple $\mathfrak{M} = (T, <, \mathcal{I})$, where $\mathcal{I}$ is a function assigning to every $t \in T$ a DL interpretation $\mathcal{I}(t) = (\Delta(t), \cdot^{\mathcal{I}(t)})$. We say that $\mathfrak{M}$ is a* model *of $\mathcal{T}$ and $\mathcal{A}_T$, whenever $\mathcal{I}(t)$ is a model of $\mathcal{T}$ and $\mathcal{A}_T(t)$, for every $t \in T$.*

5 Temporal Query Language

In this section, we define and study a novel temporal query language $\mathcal{TQL}$, or strictly speaking, a family of such languages for querying temporal ABoxes *w.r.t.* standard TBoxes. At its core, our contribution should be seen as a general mechanism for constructing practical query formalisms, based on combinations of temporal logics with certain classes of first-order queries over DLs. This mechanism can be shortly described as follows. Consider a temporal logic $\mathcal{TL}$ and a class of queries $\mathcal{Q}$. We aim at identifying a fragment of *temporal first-order logic*, based on the operators of $\mathcal{TL}$, whose first-order component coincides with the class $\mathcal{Q}$. To this end, we follow the well-studied methodology of *temporalization of logic systems*, introduced by Finger and Gabbay [6]. Essentially, $\mathcal{TQL}$ is defined as the set of all $\mathcal{TL}$-formulas whose atomic subformulas are substituted with $\mathcal{Q}$-queries. The central motivation behind such a construction is to enable decoupling the problem of answering embedded $\mathcal{Q}$-queries from reasoning in $\mathcal{TL}$, which can be both efficiently addressed by existing, specialized tools. As it turns out, some potential interactions between the Boolean operators of $\mathcal{TL}$-formulas with those of $\mathcal{Q}$-queries make such decoupling still impossible in general. Hence, as a solution, we advocate a controlled use of epistemic semantics for interpreting the embedded $\mathcal{Q}$-queries, along the lines proposed by Calvanese et al. [16]. This, as we argue in Section 5.2, leads to a desirable theoretical and practical characterization of $\mathcal{TQL}$.

In our scenario, we focus on the class of conjunctive queries, as the most popular type of queries studied in the context of DLs. As the baseline temporal language we consider *first-order monadic logic of orders* (FOMLO), which is known to be expressively complete w.r.t. all linear orders [17], and thus subsumes a number of most popular linear temporal logics, including the prominent Propositional Linear Temporal Logic (PLTL).

5.1 Syntax and semantics

To keep the design of $\mathcal{TQL}$ possibly modular, and yet maximally generic, we first introduce a mechanism of abbreviating the temporal components of the queries into customary *temporal connectives*. Those connectives, defined analogical to Chomicki and Toman [18], are used as templates to be instantiated with particular CQs and further combined by means of Boolean operators. The syntax of

FOMLO is based on a countably infinite set $\mathsf{T_V}$ of time instant variables, such that $\mathsf{T_V} \cap \mathsf{N_V} = \emptyset$, one binary predicate $<$ and a countably infinite set $\mathsf{P_V}$ of monadic predicate variables.

Definition 4 (Temporal connectives: syntax). *A ϕ-formula is an expression constructed according to the grammar:*

$$\phi \ ::= \ u < v \ \mid \ \neg\phi \ \mid \ \phi \wedge \phi \ \mid \ \forall x.\phi \ \mid \ X(u)$$

where $u, v \in T \cup \mathsf{T_V}$, $x \in \mathsf{T_V}$ and $X \in \mathsf{P_V}$. An n-ary temporal connective *is a ϕ-formula containing $k \geq 0$ free variables $x_1, \ldots, x_k \in \mathsf{T_V}$, called the* temporal answer variables*, and $n \geq 0$ predicate variables $X_1, \ldots, X_n \in \mathsf{P_V}$. We define Ω to be a finite set of temporal connectives, where each connective $\omega \in \Omega$ is given via a definition consisting of a name $\omega(\vec{x})(\vec{X})$, with $\vec{x} = x_1 \ldots, x_k$ and $\vec{X} = X_1, \ldots, X_n$, and a (definitional) ϕ-formula ω^*.*

Intuitively, the predicate variables are place holders for CQs, which we add in the next step. The temporal answer variables range over time instants, which are explicitly represented in the answers to temporal queries.[3] A small sample of possible temporal connectives is given in Figure 1. Note, that we use some common abbreviations, such as $\exists, \rightarrow, \leq$, as well as compositions $x_1 < x_2 < x_3$, whose meaning is as expected.

$$\begin{array}{rcl}
\textbf{always}(X_1) & \triangleq & \forall x_1.X_1(x_1) \\
\textbf{sometime}(X_1) & \triangleq & \exists x_1.X_1(x_1) \\
\textbf{in}(x_1)(X_1) & \triangleq & X_1(x_1) \\
\textbf{after}(x_1, x_2) & \triangleq & x_2 < x_1 \\
\textbf{during-interval}(x_1, x_2)(X_1) & \triangleq & x_1 \leq x_2 \wedge \forall x_3.(x_1 \leq x_3 \leq x_2 \rightarrow X_1(x_3)) \\
\textbf{in-since}(x_1)(X_1, X_2) & \triangleq & \exists x_2.(x_2 < x_1 \wedge X_2(x_2) \wedge \forall x_3.(x_2 < x_3 \leq \\
 & & x_1 \rightarrow X_1(x_3))) \\
\textbf{in-until}(x_1)(X_1, X_2) & \triangleq & \exists x_2.(x_1 < x_2 \wedge X_2(x_2) \wedge \forall x_3.(x_1 \leq x_3 < \\
 & & x_2 \rightarrow X_1(x_3)))
\end{array}$$

Fig. 1. Examples of temporal connectives.

The satisfaction relation is defined in terms of the standard first-order semantics, *modulo* an extra condition warranting the satisfaction of predicate variables.

Definition 5 (T-substitution). *For a time domain $(T, <)$, a T-substitution is a mapping $\pi : T \cup \mathsf{T_V} \mapsto T$ such that $\pi(t) = t$ for every $t \in T$.*

[3] In practice, the range of answer variables might need to be further restricted in order to finitize the number of possible answers. In the context of temporal databases, it is common to consider only time instants that are explicitly mentioned in the data (in our case: temporal ABox). This, however, might require certain normalization of the used time-stamps — a problem which we do not address here.

Definition 6 (Temporal connectives: satisfaction relation). *Let* $\mathfrak{M} = (T, <, \mathcal{I})$ *be a temporal model and* $\omega \in \Omega$ *a temporal connective with the definitional formula* ω^**. For a* T*-substitution* π*, the satisfaction relation* $\mathfrak{M}, \pi \models \omega^*$ *is defined inductively as follows:*

- $\mathfrak{M}, \pi \models u < v$ iff $\pi(u) < \pi(v)$,
- $\mathfrak{M}, \pi \models \neg\phi$ iff $\mathfrak{M}, \pi \not\models \neg\phi$,
- $\mathfrak{M}, \pi \models \phi \wedge \psi$ iff $\mathfrak{M}, \pi \models \phi$ *and* $\mathfrak{M}, \pi \models \psi$,
- $\mathfrak{M}, \pi \models \forall x.\phi$ iff *for every* $t \in T$ *it is the case that* $\mathfrak{M}, \pi[x \mapsto t] \models \phi$,
- $\mathfrak{M}, \pi \models X_i(u)$ iff *the CQ associated with* X_i *is satisfied in* $\mathfrak{M}, \pi(u)$ *(see Definition 8).*

where $\pi[x \mapsto t]$ *denotes a* T*-substitution exactly as* π *except for that we fix* $\pi(x) = t$*.*

Finally, we define the syntax and semantics of the temporal query language.

Definition 7 (Temporal query language: syntax). *The* temporal query language $\mathcal{TQL}$ *is induced by the following grammar:*

$$\psi \ ::= \ \omega(\vec{x})(q_1(\vec{y}_1), \ldots, q_n(\vec{y}_n)) \ \mid \ \neg\psi \ \mid \ \psi \wedge \psi$$

where $\omega \in \Omega$ *is an* n*-ary temporal connective with temporal answer variables* $\vec{x}$*, and every* $q_i(\vec{y}_i)$ *is a CQ with answer variables* $\vec{y}_i$*, for* $1 \leq i \leq n$*. We write* $\psi(\vec{x}, \vec{y})$*, to denote a* $\mathcal{TQL}$ *query* ψ *with temporal answer variables* $\vec{x}$ *and CQ answer variables* $\vec{y}$*.*

An answer to a $\mathcal{TQL}$ query is a pair of sequences of time instants from T and individual names from $\mathsf{N_I}$, which substituted for the respective temporal and CQ answer variables must satisfy the query. The answer variables of both types can be shared among different CQs and temporal connectives occurring in the query, thus facilitating descriptions of complex dependencies between temporal data (*cf.* Example 1). To formally introduce the semantics of $\mathcal{TQL}$ queries, we first fix useful notation for handling subsequences of CQ answers. Let $\vec{y} = y_1, \ldots, y_k$ be a sequence of answer variables and $\vec{a} = a_1, \ldots, a_k$ a corresponding sequence of individual names. For an arbitrary subsequence $\vec{y}' \subseteq \vec{y}$, i.e. a subset of $\vec{y}$ preserving the ordering, we write $\vec{a}|_{\vec{y}'}$ to denote the subsequence of $\vec{a}$ such that for every $1 \leq i \leq k$, a_i occurs in $\vec{a}|_{\vec{y}'}$ *iff* y_i occurs in $\vec{y}'$.

Definition 8 (Temporal query language: semantics). *Let* $\psi(\vec{x}, \vec{y})$ *be a* $\mathcal{TQL}$ *query, with* $\vec{x} = x_1, \ldots, x_k$ *and* $\vec{y} = y_1, \ldots, y_l$*. For a pair of sequences* $(\vec{t}, \vec{a})$*, where* $\vec{t} = t_1, \ldots, t_k \in T$ *and* $\vec{a} = a_1, \ldots, a_l \in \mathsf{N_I}$*, a* $(\vec{t}, \vec{a})$*-match to* ψ *in a model* $\mathfrak{M} = (T, <, \mathcal{I})$ *is a* T*-substitution* π*, such that* $\pi(x_i) = t_i$*, for every* $1 \leq i \leq k$*, and* $\mathfrak{M}, \pi \models_{\vec{a}} \psi$*, where the satisfaction relation* $\models_{\vec{a}}$ *is defined inductively as follows:*

- $\mathfrak{M}, \pi \models_{\vec{a}} \omega(\vec{x_i})(q_1(\vec{y}_1), \ldots, q_n(\vec{y}_n))$ iff $\mathfrak{M}, \pi \models \omega^*$ *(see Definition 6), where for every* $1 \leq i \leq n$ *and any* T*-substitution* π' *we set:*

$$\mathfrak{M}, \pi' \models X_i(\pi'(u)) \text{ iff } \mathcal{I}(\pi'(u)) \models q_i[\vec{a}|_{\vec{y}_i}], \qquad (\dagger)$$

– $\mathfrak{M}, \pi \models_{\vec{a}} \neg\varphi$ iff $\mathfrak{M}, \pi \not\models_{\vec{a}} \varphi$,
– $\mathfrak{M}, \pi \models_{\vec{a}} \varphi \wedge \psi$ iff $\mathfrak{M}, \pi \models_{\vec{a}} \varphi$ *and* $\mathfrak{M}, \pi \models_{\vec{a}} \psi$.

We write $\mathfrak{M} \models \psi[\vec{t}, \vec{a}]$ *whenever there exists a* $(\vec{t}, \vec{a})$*-match to* ψ *in* $\mathfrak{M}$ *and* $\mathcal{T}, \mathcal{A}_T \models \psi[\vec{t}, \vec{a}]$*, whenever there exists a* $(\vec{t}, \vec{a})$*-match to* ψ *in every model of* $\mathcal{T}$ *and* $\mathcal{A}_T$*. In the latter case* $\vec{t}, \vec{a}$ *is called a* certain answer *to* ψ *w.r.t.* $\mathcal{T}, \mathcal{A}_T$.

Example 1. We formulate a $\mathcal{TQL}$ query $\psi(x_1, x_2, y)$ requesting all patients y who have been ever diagnosed with some allergy, at some point x_1 were administered a new drug, and at some point x_2, after x_1, had symptoms of an allergic reaction. The precise meaning of the temporal connectives used in the query is as defined in Figure 1.

$$\begin{aligned}\psi(x_1, x_2, y) \;::=\; & \mathbf{sometime}(\exists x.(\mathsf{Patient}(y) \wedge \mathsf{diagnosedWith}(y, x) \wedge \mathsf{Allergy}(x))) \\ & \wedge \mathbf{in}(x_1)(\exists x.(\mathsf{administered}(y, x) \wedge \mathsf{NewDrug}(x))) \wedge \\ & \wedge \mathbf{after}(x_2, x_1) \\ & \wedge \mathbf{in}(x_2)(\exists x.(\mathsf{hasSymptom}(y, x) \wedge \mathsf{AllergicReaction}(x)))\end{aligned}$$

Consider the TBox $\mathcal{T}$ containing axioms:

$$\begin{aligned}&\mathsf{AllergicPatient} \sqsubseteq \mathsf{Patient} \sqcap \exists \mathsf{diagnosedWith}.\mathsf{Allergy} \\ &\mathsf{TestPatient} \sqsubseteq \mathsf{Patient} \sqcap \exists \mathsf{administered}.\mathsf{NewDrug}\end{aligned}$$

and the temporal ABox $\mathcal{A}$ containing time-stamped assertions:

$[1,5] : \mathsf{AllergicPatient}(\mathit{john})$	$[2,3] : \mathsf{Patient}(\mathsf{carl})$
$[1,2] : \mathsf{hasSymptom}(\mathit{john}, \mathsf{id1})$	$[1,2] : \mathsf{hasSymptom}(\mathit{carl}, \mathsf{id3})$
$[2,2] : \mathsf{AllergicReaction}(\mathsf{id1})$	$[2,2] : \mathsf{AllergicReaction}(\mathsf{id3})$
$[4,5] : \mathsf{TestPatient}(\mathsf{john})$	$[2,3] : \mathsf{diagnosedWith}(\mathsf{carl}, \mathsf{id4})$
$[6,6] : \mathsf{hasSymptom}(\mathsf{john}, \mathsf{id2})$	$[2,3] : \mathsf{Allergy}(\mathsf{id4})$
$[6,6] : \mathsf{AllergicReaction}(\mathsf{id2})$	$[5,5] : \mathsf{TestPatient}(\mathsf{carl})$

Given the time domain of natural numbers $(\mathbb{N}, <)$ there are two certain answers to the query $\psi(x_1, x_2, y)$, namely: $(4, 6, \mathsf{john})$ and $(5, 6, \mathsf{john})$.

5.2 Practical query answering

As it turns out, under the introduced semantics the expressive power of $\mathcal{TQL}$ is still too high to provide reasonable guarantees for the worst-case complexity of temporal query answering, and for the possibility of reusing the existing query answering techniques and tools. The level of interaction between the Boolean operators of the temporal language with CQs is sufficient to enable encoding Boolean combinations of conjunctive queries (BCCQs) over DLs, i.e. formulas induced by the grammar:

$$\varphi \;::=\; q \;\mid\; \neg\varphi \;\mid\; \varphi \wedge \psi.$$

where q is a CQ. The decidability of BCCQs answering over DLs is, to the best of our knowledge, an open problem. Some of the largest classes of queries whose decidability has been studied so far are in fact unions (disjunctions) of CQs [1] and their syntactic generalization — positive existential queries [19]. In

order to render query answering in $\mathcal{TQL}$ practical, we therefore need to employ some means of constraining the language. Quite a trivial fix is to tame the expressiveness of CQs, for instance by considering only CQs without existentially bounded variables — thus a variation of instance queries. In such case, the query answering can be reduced to reasoning with temporalized ABox axioms w.r.t. global TBox. As explained in [13], for a temporal logic coinciding with PLTL and an arbitrary DL with at least PSPACE-hard satisfiability problem, the complexity of the task remains the same as for the satisfiability in the underlying DL.

A much more interesting way of alleviating the problem of handling BC-CQs, however, is to restrict the level of interaction between the operators of the temporal language with those of the embedded CQs, without reducing the expressiveness of the queries. To this end we propose to apply a limited form of the Closed World Assumption (CWA). Although essentially incompatible with the open-world semantics of DLs, a controlled use of CWA is claimed to be justified and beneficial in various application scenarios related to OBDA and Semantic Web reasoning [20,16,21]. In our case, we are interested in restricting $\mathcal{TQL}$ in a way that would enable answering individual CQs under the standard semantics, but at the same time, interpreting negation in front of CQs as Negation-As-Failure, and reducing the problem of answering BCCQs to Boolean operations over the certain answers to CQs. A clean and straightforward method of achieving this effect, advocated and studied in depth in [16], is to bind every occurrence of a CQ in a $\mathcal{TQL}$ query with the *autoepistemic* $\mathbf{K}$-*operator*. Essentially, the operator $\mathbf{K}$ enforces that a bounded CQ is satisfied in a model, for a given answer, only if this answer is *known* to be certain, or formally:

$$\mathcal{I} \models \mathbf{K}q[\vec{a}] \text{ } \textit{iff} \text{ } \mathcal{T}, \mathcal{A} \models q[\vec{a}]$$

where $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$. This immediately entails the requested reductions of a limited, closed-world flavor:

$$\mathcal{T}, \mathcal{A} \models \mathbf{K}q[\vec{a}] \text{ } \textit{iff} \text{ } \mathcal{T}, \mathcal{A} \models q[\vec{a}]$$
$$\mathcal{T}, \mathcal{A} \models \neg\mathbf{K}q[\vec{a}] \text{ } \textit{iff} \text{ } \mathcal{T}, \mathcal{A} \not\models q[\vec{a}]$$
$$\mathcal{T}, \mathcal{A} \models \mathbf{K}q_1[\vec{a}] \vee \mathbf{K}q_2[\vec{b}] \text{ } \textit{iff} \text{ } \mathcal{T}, \mathcal{A} \models q_1[\vec{a}] \text{ or } \mathcal{T}, \mathcal{A} \models q_2[\vec{b}]$$

Observe that the set of certain answers to a single CQ is invariant to the possible application of the $\mathbf{K}$-operator in front of the query. Thus, the closed-world reasoning, emerging only on the level of Boolean combinations of CQs, does not affect the basic assumption of possible incompleteness of data, inherent to the OBDA paradigm.

Eventually, by replacing every q in $\mathcal{TQL}$ queries with $\mathbf{K}q$, or simply by interpreting it *as if it was bounded* by $\mathbf{K}$ (as we do below), we obtain the desired, well-behaved temporal query language.

Definition 9 ($\mathcal{TQL}$ semantics with epistemic interpretation of CQs). *The semantics of $\mathcal{TQL}$ with* epistemic interpretation *of embedded CQs is exactly the same as in Definition 8, modulo the replacement of the condition* (†) *with the following one:*

$$\mathfrak{M}, \pi' \models X_i(\pi'(u)) \text{ iff } \mathcal{T}, \mathcal{A}_T(\pi'(u)) \models q_i[\vec{a}|_{\vec{y}_i}]. \qquad (\ddagger)$$

To witness the difference between evaluating $\mathcal{TQL}$ queries under the two compared semantics, consider an example involving TBox $\mathcal{T} = \{\mathsf{A} \sqsubseteq \neg\mathsf{D},\ \mathsf{B} \sqcap \mathsf{C} \sqsubseteq \mathsf{D}\}$, temporal ABox $\mathcal{A} = \{[1,1] : \mathsf{A}(\mathsf{a}),\ [1,2] : \mathsf{B}(\mathsf{a}),\ [2,3] : \mathsf{C}(\mathsf{a})\}$ and query $\psi(x,y) \ ::= \neg\mathbf{in}(x)(\mathsf{D}(y))$. Under the original semantics, presented in Definition 8, the query returns a unique certain answer $(1,\mathsf{a})$. On the other hand, by enforcing the epistemic interpretation of the embedded CQ $q(y) ::= \mathsf{D}(y)$, as argued above, and setting the time domain of natural numbers, we obtain an infinite set of certain answers $\{(t,\mathsf{a}) \mid 2 \neq t \in \mathbb{N}\}$.

Notably, the condition $\mathcal{T}, \mathcal{A}_T(\pi'(u)) \models q_i[\vec{a}|_{\vec{y}_i}]$ in $(\ddagger)$ is an instance of the standard CQ answering problem. Moreover, it is the only point in the revised semantics where DL reasoning is intertwined with reasoning over the temporal language. What follows, is that the most natural algorithm answering $\mathcal{TQL}$ queries can be constructed by augmenting any standard decision procedure for the satisfiability in the temporal language with an oracle answering CQs over the designated snapshots of the ABox *w.r.t.* the TBox. As the decision problem in FOMLO is known to be PSPACE-complete [17], we thus obtain a result on the combined complexity of answering $\mathcal{TQL}$ queries.

Theorem 1 (Combined complexity of $\mathcal{TQL}$ query answering). *Let ψ be a $\mathcal{TQL}$ query over a temporal ABox $\mathcal{A}_T$ w.r.t. TBox $\mathcal{T}$, where ABox and TBox axioms are expressed in a DL language $\mathcal{L}$. The combined complexity of deciding $\mathcal{T}, \mathcal{A}_T \models \psi[\vec{t},\vec{a}]$, for a pair of sequences $\vec{t},\vec{a}$, under the epistemic interpretation of the CQs embedded in ψ, is* $\text{PSPACE}^{QA(\mathcal{L})}$*-complete, where $QA(\mathcal{L})$ is an oracle answering CQs in $\mathcal{L}$.*

This seemingly unsurprising result has some significant theoretical and practical implications. On the theoretical side, it guarantees that answering $\mathcal{TQL}$ queries under the epistemic interpretation of CQs remains decidable, as long as answering CQs over the respective DLs is decidable. Moreover, it establishes a bridge for an immediate transfer of the combined complexity results. For instance, when $\mathcal{L} = \mathcal{ALC}$, answering $\mathcal{TQL}$ queries is $\text{PSPACE}^{\text{EXPTIME}}$-complete, thus effectively EXPTIME-complete, as the combined complexity of CQ answering in $\mathcal{ALC}$ is EXPTIME-complete [22]. Analogically, for $\mathcal{L} = \mathcal{SHIQ}$, the problem is $\text{PSPACE}^{\text{2EXPTIME}}$-complete, and effectively 2EXPTIME-complete. In general, the combined complexity of answering $\mathcal{TQL}$ queries for an arbitrary DL $\mathcal{L}$ is equal to the complexity of answering CQs in $\mathcal{L}$, provided that the latter problem is at least PSPACE-hard. This observation naturally generalizes over query languages based on combinations of FOMLO with arbitrary classes of first-order queries. Whenever the complexity of answering $\mathcal{Q}$-queries over $\mathcal{L}$, for a given query class $\mathcal{Q}$ and a DL $\mathcal{L}$, is at least PSPACE-hard then answering the resulting temporal queries over $\mathcal{L}$ remains in the same complexity class. Otherwise it is PSPACE-complete. This demonstrates that the temporalization technique employed here yields computationally cheap, yet expressive, temporal query languages over temporal ABoxes. In fact, temporalization of query languages for expressive DLs, subsuming $\mathcal{ALC}$, comes for free.

From the practical perspective, the restricted interaction between the temporal component and CQs, reflected in Theorem 1, promises relatively straight-

forward implementations of $\mathcal{TQL}$ query engines based on existing technologies, e.g.: temporal theorem provers and CQ answering tools. Roughly, to determine whether a candidate answer to a query ψ is certain for $\mathcal{T}, \mathcal{A}_T$, it suffices to check whether the direct rendering of ψ into FOMLO is satisfiable, where every CQ embedded in ψ is seen as a predicate variable, whose truth-value in a given time instant is determined by a call to an external CQ answering tool over the respective snapshot of $\mathcal{A}_T$ *w.r.t.* $\mathcal{T}$.[4]

Some further interesting prospects concern answering $\mathcal{TQL}$ queries over the DL-Lite family of DLs, enjoying the FO-rewritability property [2]. It is known that CQ answering in DL-Lites can be carried out efficiently using highly optimized RDBMSs. In a nutshell, for TBox $\mathcal{T}$, ABox $\mathcal{A}$ and a CQ q, one can always find a first-order query $q^{\mathcal{T}}$, such that for every $\vec{a}$ it is the case that $\mathcal{T}, \mathcal{A} \models q[\vec{a}]$ *iff* $\mathcal{A} \models q^{\mathcal{T}}[\vec{a}]$, where the latter problem can be solved directly in an RDBMS. Clearly, an analogical approach should enable rewriting a $\mathcal{TQL}$ query over $\mathcal{T}, \mathcal{A}_T$ into a temporal first-order formula, which could be then efficiently encoded and evaluated as a TSQL2 query [5] over a temporal database $\mathcal{A}_T$. Although providing precise definition of such a translation and proving its correctness is left as future work, we expect it to be straightforward given that every $\mathcal{TQL}$ query corresponds to a temporal formula with embedded CQs, where each CQ q can be replaced with the corresponding first-order formula $q^{\mathcal{T}}$ obtained by means of established FO-rewritability techniques.

6 Discussion and Outlook

The design of the language $\mathcal{TQL}$ follows closely the principles of query languages for temporal databases, as outlined in e.g. [18]. In the general TDB setup, the query component $\mathcal{Q}$ is based on the full first-order logic, while temporal operators defined in FOMLO can be nested within each other. Hence, the resulting language is expressively equivalent to the temporal first-order logic. In $\mathcal{TQL}$, we are deliberately constraining the query component and disallow nesting of operators in order to enable practical decoupling of the DL-level from the temporal-level reasoning. In the context of the SW, similar approaches have been proposed to deal with time-stamped RDF data [10,12] and OWL axioms [12]. Both contributions, however, lack the generality of our proposal. The temporal component of the query languages is in both cases highly restricted in order to ensure finite answer sets. In particular, Motik [12] introduces a specially fixed number of most practical temporal operators that can be combined with the data-level queries. All these can be easily restated in FOMLO, and so the language of [12] can be easily defined using the $\mathcal{TQL}$ mechanism.

An interesting open challenge is a potential integration of the framework with two orthogonal approaches to managing temporal information in the context of

[4] For time domains based on natural numbers and integers, FOMLO formulas can be translated into PLTL [17], and thus decided using off-shelf PLTL provers, such as listed in `http://www.cs.man.ac.uk/~schmidt/tools/`. CQ answering in selected DLs is supported by such systems as QuOnto, REQUIEM, Presto.

DLs, mentioned in Section 2, i.e. SW temporal vocabularies and temporal DLs. As argued below, our choice of standard temporal semantics and logic-based query formalism, renders such prospects quite realistic. A remarkable feature, shared by the two potential extensions, is that unlike the current framework, they aim to support reasoning with incomplete data, where the incompleteness occurs in the temporal dimension. Such characteristic might be conceptually attractive considering the open-world philosophy of DLs.

Supporting temporal vocabularies and semantic annotations The practice of representing and reasoning with temporal information on the Semantic Web, for instance in the field of health care support [23], suggests that the presented data model and query language might not be sufficiently flexible for real-life applications. In particular, we are incapable of:

- directly expressing typical temporal patterns occurring in temporal queries and constraints used in clinical applications, such as:
  - Visit 17 must occur *at least 1 week but no later than 4 weeks after the end of 2003 ragweed season.*
  - Administer Rapamune *1 week from* Visit 0 *daily for 84 days.*
- supporting semantic annotations whose meaning could be described in terms of rich temporal vocabularies, e.g. for a temporal assertion $\tau : \alpha$:
  - τ is a time interval from 15.05.2005 until some day on 06.2005,
  - τ is a periodic interval, of duration 5 hours, recurring 5 times every 10 hours, starting some time on 12.05.2008.

In practice, such functionalities are supported by ad-hoc architectures, which retrieve temporal information encoded in OWL-based time ontologies and process it with application-specific tools, thus sacrificing some theoretical rigor and formal transparency of provided inferences [9]. Within our framework, a natural solution to this problem is to extend the temporal language with additional predicates (e.g. involving periodicity constraints [24]) to enable ontological-style axiomatization of the background knowledge about the underlying time domain *per se*, e.g. the Gregorian calendar. Such predicates could be then meaningfully used and reasoned with on the query- and annotation-level. Such a philosophy motivates to a great extent the framework proposed by Zimmerman et al. [11]. There, however, the annotation language is a non-standard, task-specific formalism, which cannot be directly translated into temporal logics or OWL.

Integration with TDLs The framework studied in this paper is focused on querying temporal data with respect to a fixed, time-invariant terminology. A natural extension to this approach is to introduce means of querying temporal ABoxes in presence of temporal constraints occurring on the intensional, terminological level. Temporal DLs are a family of two-dimensional DLs, developed intensively in the recent years [13], intended specifically for the representation of this kind of terminologies. By allowing operators of temporal logics to occur in DL concepts, TDLs enable, for instance, to express the following axiom:

$$\mathsf{Patient} \sqcap \exists \mathsf{diagnosedWith}.\mathsf{Allergy} \sqsubseteq \mathsf{AllergicPatient}\ \mathcal{U}\ \forall \mathsf{diagnosedWith}.\neg\mathsf{Allergy}$$

The axiom states that whenever a patient is diagnosed with an allergy, she should be considered an allergic patient until ($\mathcal{U}$) she is diagnosed with no allergies. Interestingly, TDL TBoxes are interpreted over the same type of semantic structures as used in our framework, i.e. tuples $\mathfrak{M} = (T, <, \mathcal{I})$. This means, that from the formal perspective integration of $\mathcal{TQL}$ with TDLs can be achieved seamlessly. Obviously, query answering in such setting should likely be computationally more expensive, considering that already the satisfiability problem in TDLs is usually harder than in the underlying DLs. So far the only query language for TDLs has been proposed by Artale et al. [14]. Differently than here, the queries are defined as unions of CQs, where the atomic predicates can be possibly preceded by temporal operators. As a consequence, the reuse of existing CQ answering techniques is not directly possible within this approach.

7 Conclusions

We believe that the framework proposed in this paper marks a first promising step towards establishing a generic approach to representing and querying temporal data under DL ontologies. Naturally, a number of important problems, which we merely touched upon, are left open to future research. Among others, it is critical to conduct a systematic study of possible ways of restricting the $\mathcal{TQL}$-like languages, in order to turn the query answering problem feasible in practice. The use of epistemic semantics, suggested here, is only one of possible options. Other might involve more fine-grained constraints on the expressiveness of the temporal component, the first-order component or both. We also advocate a study of possible extensions of the framework towards integration with temporal DLs and rich SW temporal vocabularies.

References

1. Glimm, B., Horrocks, I., Lutz, C., Sattler, U.: Conjunctive query answering for the description logic SHIQ. In: Journal of Artificial Intelligence Research. (2008)
2. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. of Automated Reasoning **39**(3) (2007) 385–429
3. Chomicki, J., Toman, D.: Temporal Databases. In: Handbook of Temporal Reasoning in Artificial Intelligence (Foundations of Artificial Intelligence), Elsevier Science Inc. (2005) 429–468
4. Montanari, A., Chomicki, J.: Time domain. In: Encyclopedia of Database Systems. (2009) 3103–3107
5. Böhlen, M.H., Chomicki, J., Snodgrass, R.T., Toman, D.: Querying TSQL2 Databases with Temporal Logic. In Apers, P.M.G., Bouzeghoub, M., Gardarin, G., eds.: EDBT. Volume 1057 of Lecture Notes in Computer Science., Springer (1996) 325–341
6. Finger, M., Gabbay, D.M.: Adding a temporal dimension to a logic system. Journal of Logic, Language and Information **1**(3) (1992) 203–233

7. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. ACM Trans. Asian Lang. Inf. Process. **3**(1) (2004) 66–85
8. Batsakis, S., Stravoskoufos, K., Petrakis, E.G.M.: Temporal Reasoning for Supporting Temporal Queries in OWL 2.0. In König, A., Dengel, A., Hinkelmann, K., Kise, K., Howlett, R.J., Jain, L.C., eds.: KES (1). Volume 6881 of Lecture Notes in Computer Science., Springer (2011) 558–567
9. O'Connor, M.J., Das, A.K.: A method for representing and querying temporal information in OWL. Biomedical Engineering Systems and Technologies (Selected Papers) (2011) 97–110
10. Gutierrez, C., Hurtado, C.A., Vaisman, A.A.: Introducing Time into RDF. IEEE Trans. Knowl. Data Eng. **19**(2) (2007) 207–218
11. Zimmermann, A., Lopes, N., Polleres, A., Straccia, U.: A general framework for representing, reasoning and querying with annotated Semantic Web data. Web Semantics: Science, Services and Agents on the World Wide Web **11** (2012) 72–95
12. Motik, B.: Representing and querying validity time in RDF and OWL: A logic-based approach. Web Semantics: Science, Services and Agents on the World Wide Web **12-13** (2012) 3–21
13. Lutz, C., Wolter, F., Zakharyaschev, M.: Temporal description logics: A survey. In: Proc. of TIME-08. (2008)
14. Artale, A., Franconi, E., Wolter, F., Zakharyaschev, M.: 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)
15. Baader, F., Calvanese, D., Mcguinness, D.L., Nardi, D., Patel-Schneider, P.F.: The description logic handbook: theory, implementation, and applications. Cambridge University Press (2003)
16. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Eql-lite: Effective first-order query processing in description logics. In: Proc. of IJCAI-07. (2007)
17. Reynolds, M.: The complexity of decision problems for linear temporal logics. Journal of Studies in Logic **3**(1) (2010) 1917
18. Chomicki, J., Toman, D.: Temporal logic in information systems. In Chomicki, J., Saake, G., eds.: Logics for Databases and Information Systems, Kluwer (1998) 31–70
19. Ortiz, M., Calvanese, D., Eiter, T.: Data complexity of query answering in expressive description logics via tableaux. J. of Automated Reasoning **41** (2008) 61–98
20. Grimm, S., Motik, B.: Closed world reasoning in the semantic web through epistemic operators. In: Second International Workshop on OWL: Experiences and Directions (OWLED 2006). (2005)
21. Lutz, C., Seylan, I., Wolter, F.: Mixing open and closed world assumption in ontology-based data access: Non-uniform data complexity. In: Proc. of the International Workshop on Description Logics (DL2012). (2012)
22. Lutz, C.: The complexity of conjunctive query answering in expressive description logics. In: Proc. of IJCAR-08. Number 5195 in LNAI (2008) 179–193
23. Shankar, R.D., Martins, S.B., O'Connor, M.J., Das, A.K.: An ontological approach to representing and reasoning with temporal constraints in clinical trial protocols. In: HEALTHINF (1), INSTICC - Institute for Systems and Technologies of Information, Control and Communication 87–93
24. Demri, S.: LTL over integer periodicity constraints. Theoretical Computer Science **360** (August 2006) 96–123