Reasoning with Temporal ABoxes: Combining DL-L*ite*$_{core}$ with CTL

Francesco Pagliarecci, Luca Spalazzi, and Gilberto Taccari

DII – Università Politecnica delle Marche,
via Brecce Bianche, I-60131 Ancona, Italy.
{pagliarecci,spalazzi,g.taccari}@dii.univpm.it

Abstract. The work shows the combination of standard description logics (DLs) with standard temporal logics (TLs). Indeed, the introduction of DLs as logic-based knowledge representation formalisms has emerged in many fields and many applications have taken advantage from it. Although that, in many of these applications also temporal aspects play an important role. It follows that a new formalism is needed in order to represent both terminological and temporal knowledge. At this aim, the majority of research works proposed the combination of DLs and TLs producing a new formalism to knowledge representation: temporal description logics (TDLs). This work illustrates the combination of the DL-L*ite*$_{core}$ description logic with the temporal logic CTL. It defines a temporal knowledge base with time-invariant TBox and time-variant ABox. Furthermore an algorithm based on semantic model checking is proposed in order to query the knowledge base.

Key words: Description Logics, Temporal Logics, Semantic Web, DL-L*ite*, CTL, Model Checking.

1 Introduction

Recent work in service oriented architectures, business processes, databases focused on the need of dealing both with facts and processes [1–10]. In particular, the Temporal Description Logic is becoming more and more used for such a kind of problems [1, 4, 8–10]. Nevertheless, reasoning and query answering for a Temporal Description Logic is, in general, untractable or even undecidable (depending on which kind of Descrption Logic and which kind of Temporal Logic are considered [11]). This is due to the fact that in general a temporal knowledge base consists of a time-variant TBox and a time-variant ABox, the ABox is defined according to an intensional definition, with a time-variant domain, under the Open World Assumption. As a consequence, several authors studied how complexity can be reduced for specific types of Temporal Description Logics where some of the above assumptions are dropped [11].

This paper presents a *minimalistic* approach based on a temporal knowledge base where the TBox is time invariant, the ABox is defined in an extensional fashion, the domain is constant, and the Closed World Assumption is adopted.

The above choices allows us to obtain a tractable query answering algorithm that can be still used in several problems of practical interest, e.g. selection and composition of semantic web services, verification of business processes, management of e-learning processes.

The paper is structured as follows. Section 2 shows works that deal with temporal description logics. Section 3 depicts the temporal knowledge base proposed in this work. Section 4 illustrates the syntax and the semantics of temporal conjunctive queries used to query the temporal knowledge base. In 5 is shown the temporal conjunctive query answering algorithm based on propositional model checking. Finally, Section 6 closes the paper discussing the obtained results and some comparisons with other works are given. Finally, it will be proposed the future work.

2 Related Work

In the recent years many works focused on Temporal Description Logics [12, 11]. They mostly inspect how terminological aspects can be combined with temporal ones and evaluate the complexity of reasoning with such logics based on two-dimensional semantics [13]. Furthermore, other works introduce temporal description logics in order to deal with particular knowledge base query problems in several application fields.

Some works deal with web services specially for discovery, selection and composition[1, 14–16, 4]. Agarwal [1, 14] integrates the description logics $\mathcal{SHIQ}$(D) and $\mathcal{SHOIN}$(D) with the temporal description logic *μ-calculus*. The goal of the work is the definition of a specification language for discovery, selection and composition of web services. The works proposed by Pistore *et al.* [15] and Di Pietro *et al.* [16, 4] deal with semantic web services selection using semantic model checking over the behavior description of services.

Weitl *et al.* [17] integrate the description logic $\mathcal{ALC}$ with the temporal logic CTL. in order to automate the verification of semi-structured documents. In that work, each proposition that appears in a CTL formula is a description logic axiom, i.e. a TBox statement. In a further work [18], they deal also with the generation of a more precise and comprehensible couterexample than the previous algorithm.

Hariri *et al.* [8] introduce the so-called Knowledge and Action Bases (KABs). A KAB is a knowledge base composed by a time invariant TBox expressed in DL-L*ite* and a variable ABox. Starting from an initial state of the knowledge base ABox, its consecutive states are determined by executing conditioned actions over the ABox. KAB properties verification is done using the μ-calculus temporal logic enriched with conjunctive queries. Baader *et al.* [9] develop a mechanism to temporalizing Ontology-Based Data Access (OBDA). They define a temporal knowledge base composed by a time invariant TBox and a sequence of ABoxes that describes the observations about the state of a system of interest. Such knowledge base can be interrogated with temporal specifications based on conjunctive queries and LTL temporal logic.

3 Temporal Knowledge Base Representation

In general, a temporal knowledge base can be defined as follows:

Definition 1 (Temporal Knowledge Base). *A* Temporal Knowledge Base $\mathcal{K} = \langle \mathcal{T}_i, \mathcal{A}_i \rangle_{i \geq 0}$ *consists of a sequence of tuples* $\langle \mathcal{T}_i, \mathcal{A}_i \rangle$ *which vary in respect to time instants* $i \geq 0$*, where:* $\mathcal{T}_i$ *and* $\mathcal{A}_i$ *are respectively the terminological part (*TB*ox) and the assertional part (*AB*ox) of the knowledge base at instant* i*.*

In literature [12, 11] many approaches are proposed in order to specialize the general definition of temporal knowledge base. Several of them focus on the definition of temporal description logics and, consequently, accomplish specialized knowledge bases suitable for the logics. The main elements that can be vary in order to realize different temporal knowledge bases are the following. The domain $\Delta^{\mathcal{I}}$ of the *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ may be constant or vary over the time. The closed or open world assumption which determines whether the information available is complete or not. Furthermore both TBoxes and ABoxes may vary over the time. Finally, concerning the assertional part of the knowledge base, ABoxes evolution can be intesionally or extensionally defined. Those temporal knowledge base design choices influence the knowledge base expressiveness and complexity. In the following are shown and explained the choices taken for this work.

First of all, it is supposed to have a constant (*constant domain assumption* [11]) and finite domain and the knowledge base makes use of the *closed world assumption*. This means that the domain does not depend on time instant and that the available information is complete. The above assumptions make feasible model checking as described in section 5.

According to Definition 1, a temporal knowledge base is formed by two components: TBox (the terminological part, i.e. a set of concepts) and ABox (the assertional part, i.e. a set of facts). For the purpose of this work, it is taken account of DL-Lite$_{core}$ [19–21].

In this description logic TBox statements are restricted only to *inclusion statements* (see Table 1.b). Intuitively, we can model an ontology only by means of subsumption between concepts, i.e. $Cl \sqsubseteq Cr$. Notice that the TBox statement $Cl_1 \sqcup Cl_2 \sqsubseteq Cr$ is equivalent to the pair of statements $Cl_1 \sqsubseteq Cr$ and $Cl_2 \sqsubseteq Cr$, and that $Cl \sqsubseteq Cr_1 \sqcap Cr_2$ is equivalent to $Cl \sqsubseteq Cr_1$ and $Cl \sqsubseteq Cr_2$. In this very simple but still expressive logic, concepts and roles are defined as follows:

$$\begin{aligned} Cl &\rightarrow A \mid \exists R \\ Cr &\rightarrow A \mid \exists R \mid \neg A \mid \neg \exists R \\ R &\rightarrow P \mid P^{-} \end{aligned}$$

Concerning an ABox, intuitively it may represent the data of a knowledge base. An ABox contains statements (or assertions) as described in Table 1.c. Intuitively, a *concept assertion a:A* (or $A(a)$) means that the individual denoted by constant a is an instance of concept A. A *role assertion* $a.P = b$ (or $P(a, b)$)

Table 1. Syntax and semantics of a DL-Lite_{core} Description Logic.

(a) Concepts and Roles.

Constructor Name	Syntax	Semantics
concept name	A	$I(A) \subseteq \Delta^{\mathcal{I}}$
role name	P	$I(P) \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
inverse role	P^-	$I(P^-) = \{(o_2, o_1) \mid (o_1, o_2) \in I(P)\}$
unqualified existential role quantification	$\exists R$	$I(\exists R) = \{o \mid \exists\, o'.(o, o') \in I(R)\}$
concept negation	$\neg A$	$\Delta^{\mathcal{I}} \setminus I(A)$
unqualified existential role quantification negation	$\neg(\exists R)$	$I(\neg\exists R) = \Delta^{\mathcal{I}} \setminus I(\exists R)$

(b) TBox (Terminology) Statements.

Statement Name	Syntax	Semantics
Concept Specification	$Cl \sqsubseteq Cr$	$I(Cl) \subseteq I(Cr)$

(c) ABox Statements.

Statement Name	Syntax	Semantics
Concept Membership	$a : A$	$I(a) \in I(A)$
Role Membership	$a.P = b$	$(I(a), I(b)) \in I(P)$

where A is a concept name, C, Cl, Cr are concepts,
P is a role name, and a, b are individuals.

means that the pair of individuals (a, b) is an instance of role P, in other words a given individual b is a value for the role P of a.

The conjunctive query answering problem has a polynomial time complexity with this kind of description logic [19].

Concerning the representation of temporal aspects, the knowledge base presented here uses *temporal* ABoxes (according to the definition in [11]), i.e. a time-invariant TBox and a time-variant ABox. Therefore an ABox represents the state of a temporal knowledge base in a precise temporal instant. A time-variant ABox can be represented according to either an intensional definition (e.g. [8]) or an estensional definition. In order to make feasible the query answering problem, in the work proposed in this paper, an estensional definition is adopted.

Borrowing the notions of *state* and *state transitions* from the formal methods field [22, 23], the temporal aspect of the knowledge base can be modeled as a state transition system on which, for each state, an ABox is defined. Indeed, the state of a system can usually be described by the values of a set of variables. These values change in the course of time and the changes are called state transitions. Some states are marked as *initial states* in order to represent the values that the system variables may have at the beginning (at time zero). These models usually assume that each state contains sufficient information to allow future behavior to be determined only by the current state, and not by its past history; this is an example of the so-called *Markovian process*. In this work, a kind of state transition system is proposed which is called an *annotated state transition system* (ASTS). A *transition relation* describes how the state can evolve, therefore which

ABoxes at instant $(t+1)$ can be reached from an ABox at instant t. Formally speaking:

Definition 2 (Annotated State Transition System). *Let $\mathcal{T}$ be the terminological part (*TBox*) of a description logic. An* annotated state transition system *Σ^A defined over a state transition system Σ is a tuple $\langle \mathcal{T}, \Sigma, \Lambda \rangle$ where:*
- *$\mathcal{T}$ is the terminological part (*TBox*) of a description logic,*
- *$\Sigma = \langle \mathcal{S}, \mathcal{S}^0, \mathcal{R} \rangle$ is the state transition system,*
- *$\mathcal{S}$ is the finite set of states;*
- *$\mathcal{S}^0 \subseteq \mathcal{S}$ is the set of initial states;*
- *$\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ is the transition relation;*
- *$\Lambda : \mathcal{S} \to 2^{\mathcal{A}_\mathcal{T}}$ is the annotation function, where $\mathcal{A}_\mathcal{T}$ is the set of all the concept assertions and role assertions defined over $\mathcal{T}$.*

As a result of the formalization of the temporal aspects of the temporal knowledge base follows the following definition which details Definition 1:

Definition 3 (Temporal ABoxes Knowledge Base). *Let $\Sigma^A = \langle \mathcal{T}, \Sigma, \Lambda \rangle$ be an* ASTS*. Let $\mathcal{K} = \langle \mathcal{T}_i, \mathcal{A}_i \rangle_{i \geq 0}$ a temporal knowledge base. Then $\mathcal{K}$ is a* Temporal ABoxes Knowledge Base *if and only if:*
- *$\forall i\ \mathcal{T}_i = \mathcal{T}$,*
- *$\forall i\ \exists\, s_i \in \mathcal{S}$ such that $\mathcal{A}_i = \Lambda(s_i)$.*

The temporal knowledge base evolution can be seen as (potentially unlimited) sequences of tuples $\langle \mathcal{T}, \Lambda(s_i) \rangle$ where $\Lambda(s_i)$ is the ABox on the state s_i at time instant i of the state transition system $\Sigma = \langle \mathcal{S}, \mathcal{S}^0, \mathcal{R} \rangle$.

4 Integrating DL-Lite with CTL

As proposed by Baader *et al.* [9] and extending the definition of conjunctive query of description logics, the *Temporal Conjunctive Query* can be formally defined as follows:

Definition 4 (Temporal Conjunctive Query). *Let Σ^A be an* ASTS*. Let $\mathcal{K}$ be a temporal* ABox*es knowledge base defined over Σ^A. Then, a* temporal conjunctive query *ϕ which refers to $\mathcal{K}$ is recursively defined as follows:*

$$\begin{aligned}\phi = q \mid \phi \wedge \phi \mid \phi \vee \phi \mid \neg\phi \mid \phi \to \phi \mid \\ \mathsf{AF}\ \phi \mid \mathsf{AG}\ \phi \mid \mathsf{EF}\ \phi \mid \mathsf{EG}\ \phi \mid \mathsf{AX}\ \phi \mid \mathsf{EX}\ \phi \mid \\ \mathsf{A}\ (\phi\ \mathcal{U}\ \phi) \mid \mathsf{E}\ (\phi\ \mathcal{U}\ \phi) \mid \mathsf{A}\ (\phi\ \mathcal{R}\ \phi) \mid \mathsf{E}\ (\phi\ \mathcal{R}\ \phi)\end{aligned}$$

where q is a conjunctive query.

Intuitively, the temporal conjunctive query is a combination of conjunctive queries tied together by means of both propositional and temporal operators. Each conjunctive query q has the form $q = \wedge_i p_i(\overline{x_i})$, where: $p_i(\overline{x_i})$ is either $x_i : C_i$ or $x_{i,1}.R_i = x_{i,2}$; and $x_i, x_{i,1}, x_{i,2}$ are variables or individuals. Assuming that $\mathsf{V}(q)$ denotes the set of variables of q and $\mathsf{C}(q)$ denotes the set of individuals of q, then $\mathsf{VC}(q) = \mathsf{V}(q) \cup \mathsf{C}(q)$ denotes the set of variables and individuals of q. When $\mathsf{V}(q) = \emptyset$, q is a *ground conjunctive query*, i.e. it is a conjunction of assertions. The definition of V is extended to temporal conjunctive queries as follows: $\mathsf{V}(\phi(q_1, \ldots, q_m)) = \mathsf{V}(q_1) \cup \mathsf{V}(q_2) \cup \ldots \mathsf{V}(q_m)$. The definition of C and VC can be extended in a similar way. When $\mathsf{V}(\phi(q_1, \ldots, q_m)) = \emptyset$, $\phi(q_1, \ldots, q_m)$ is a *ground temporal conjunctive query*. Obviously, it is possible to annotate other temporal languages (e.g. LTL [24], or μ-calculus [25]), but for the sake of complexity, the annotation of CTL formulas is enough.

The semantics of a temporal conjunctive query is inductively defined in two steps:
— first, the semantics of a conjunctive query is the base case;
— second, the semantics of a temporal conjunctive query is the inductive step based on the semantics of CTL.

The semantics of a conjunctive query is based on the semantics of the underlying description logic, as reported in [26]. This means that the semantics of a conjunctive query can be viewed in terms of tables of a relational database. $q(\overline{x})$ is the schema of the table (i.e. the relation schema) and $I(q(\overline{x}), s)$ is an instance of the table (i.e. a relation instance), in other words a set of tuples that hold in state s. Each tuple $\overline{a}$ is an assignment to $\overline{x}$ that answers to query q in the ABox $\mathcal{A}(s)$.

CTL is a propositional, branching-time, temporal logic that has Kripke semantics [24]. Intuitively, a temporal condition must be verified along all possible computation paths (state sequences) starting from the current state. Formally: *a* ***computation path*** *is an infinite sequence* $\langle s_0, s_1, s_2, \ldots, \rangle$ *of states in* $\mathcal{S}$ *such that* $\forall i. \exists\, \alpha : s_i \xrightarrow{\alpha} s_{i+1} \in \mathcal{R}$. Concerning the temporal operators (i.e., AF, EF, AX, and so on), they maintain the same intuitive meaning that they have in standard CTL. Here, the only difference is that there are conjunctive queries instead of propositions.

As a consequence, the semantics of a temporal conjunctive query can be defined as follows:

Definition 5 (Semantics of a Temporal Conjunctive Query). *Let $\mathcal{K}$ be a temporal ABoxes knowledge base defined over Σ^A. Let $q[\overline{x}]$ be a conjunctive query over $\mathcal{K}$ such that $\overline{x} = \{x_1, \ldots, x_n\}$. Let ϕ and ψ be temporal conjunctive queries. Then:*
- $\mathcal{K}, s \models q[\overline{x}]$ *iff* $\exists\, \overline{a}. I(\overline{a}) \in (\Delta^{\mathcal{I}})^n$ *and* $\langle \mathcal{T}, \mathcal{A}(s) \rangle \models q[\overline{x}/\overline{a}]$
- $\mathcal{K}, s \models \phi \wedge \psi$ *iff* $\mathcal{K}, s \models \phi$ *and* $\mathcal{K}, s \models \psi$
- $\mathcal{K}, s \models \neg\phi$ *iff* $\mathcal{K}, s \not\models \phi$
- $\mathcal{K}, s \models \mathsf{EX}\, \phi$ *iff* $\exists \langle s_i, s_{(i+1)}, s_{(i+2)}, \ldots, \rangle$ *such that* $s = s_i$ *and* $\mathcal{K}, s_{(i+1)} \models \phi$
- $\mathcal{K}, s \models \mathsf{EG}\, \phi$ *iff* $\exists \langle s_i, s_{(i+1)}, s_{(i+2)}, \ldots, \rangle$ *such that* $s = s_i$ *and* $\forall j \geq 0\ .\ \mathcal{K}, s_{(i+j)} \models \phi$
- $\mathcal{K}, s \models \mathsf{E}\, (\phi\ \mathcal{U}\ \psi)$ *iff* $\exists \langle s_i, s_{(i+1)}, s_{(i+2)}, \ldots, \rangle$ *such that* $s = s_i$ *and* $\exists j. (\mathcal{K}, s_{(i+j)} \models \psi$ *and* $\forall k < j. \mathcal{K}, s_{(i+k)} \models \phi)$

Notice that the *constant domain assumption* has been adopted, i.e. individuals are never destroyed or created over time [11] ($\Delta^{\mathcal{I}}$ is always the same for each state). The semantics of the other operators (i.e. $\vee$, $\rightarrow$, EF, AF AG, EG, AX, EX, $\mathsf{E}(.\mathcal{R}.)$, $\mathsf{A}(.\mathcal{U}.)$, $\mathsf{A}(.\mathcal{R}.)$) can easily be derived from the semantics of the minimal set of operators by means of the following equivalences:

$\mathsf{A}(\phi\,\mathcal{U}\psi) \equiv \neg\mathsf{E}(\neg\psi\mathcal{U}(\neg\phi \wedge \neg\psi)) \wedge \neg\mathsf{EG}\neg\psi$
$\mathsf{AF}\phi \equiv \neg\mathsf{EG}\neg\phi$ $\mathsf{EF}\phi \equiv \mathsf{E}(\text{true}\,\mathcal{U}\phi)$
$\mathsf{AG}\phi \equiv \neg\mathsf{EF}\neg\phi$ $\mathsf{E}(\phi\mathcal{R}\psi) \equiv \neg\mathsf{A}(\neg\phi\mathcal{U}\neg\psi)$
$\mathsf{AX}\phi \equiv \neg\mathsf{EX}\neg\phi$ $\mathsf{A}(\phi\mathcal{R}\psi) \equiv \neg\mathsf{E}(\neg\phi\mathcal{U}\neg\psi)$

5 Temporal Conjunctive Query Answering

The temporal conjunctive query answering involves two problems: the entailment problem and the answering problem.

Regarding the first problem, the following definition is given:

Definition 6 (Entailment Problem). *Let ϕ be a temporal conjunctive query. Let $\mathcal{K}$ be a temporal* ABoxes *knowledge base defined over Σ^A. The* Entailment Problem *consists in deciding whether the temporal* ABoxes *knowledge base $\mathcal{K}$ satisfies the temporal conjunctive query ϕ. Formally:*

$$\mathcal{K} \models \phi \textit{ iff } \exists\, s \in \mathcal{S}, \exists\, \overline{a}\,.\, I(\overline{a}) \in (\Delta^{\mathcal{I}})^n \textit{ and } \mathcal{K}, s \models \phi[\overline{x}/\overline{a}]$$

In other words, the entailment problem aims at checking whether a temporal knowledge base verify a given temporal specification or not. As a consequence, the result of this problem is a boolean value: *true* or *false*. The entailment problem does not give information about the temporal knowledge base data. Indeed, in most of the cases the goal of querying a knowledge base is to retrieve data that satisfy a given query. This problem can be defined as follows:

Definition 7 (Answering Problem). *Let ϕ be a temporal conjunctive query. Let $\mathcal{K}$ be a temporal* ABoxes *knowledge base defined over Σ^A. The* Answering Problem *aims at searching all the individuals in the temporal knowledge base $\mathcal{K}$ which satisfies the temporal conjunctive query ϕ. Formally:*

$$Ans(\mathcal{K}, \phi[\overline{x}]) = \{\overline{a} \mid I(\overline{a}) \in (\Delta^{\mathcal{I}})^n \textit{ and } \exists\, s \in \mathcal{S}^0, \mathcal{K}, s \models \phi[\overline{x}/\overline{a}]\}$$

where: $\overline{x}$ is an n-tuple denoting the variables of the temporal conjunctive query and $\overline{a}$ is an n-tuple of individuals that can be assigned to $\overline{x}$.

It is clear that the Answering Problem (Definition 7) is tied with the Entailment Problem (Definition 6): the former checks whether a temporal conjuctive query models or not a temporal ABoxes knowledge base and the latter gives the instances tuples which satisfies the query. It follows the proposition:

Proposition 1 (Temporal Conjunctive Query Answering). *Let $\mathcal{K}$ be a temporal* ABoxes *knowledge base defined over Σ^A. Let ϕ be a temporal conjunctive query. Let $Ans(\mathcal{K}, \phi[\overline{x}])$ be the outcome of the answering problem resulting by querying the temporal knowledge base $\mathcal{K}$ with the temporal conjunctive query ϕ. Then:*

$$\mathcal{K} \models \phi \textit{ iff } Ans(\mathcal{K}, \phi[\overline{x}]) \neq \emptyset$$

In order to deal with both the problems defined above, this work takes advantage of the work by Di Pietro *et al.* [16, 4] regarding semantic model checking for the semantic web service selection. They proposed four steps in order to verify whether a temporal specification matches a service process or not. Their algorithm can be used to deal with the temporal knowledge base proposed in this work.

Definition 8 (Semantic Model Checking Result). *Let Σ^A be an* ASTS. *Let $s \in \mathcal{S}$ be an* ASTS *state. Let $\phi[\overline{x}]$ be a temporal conjunctive query and $\phi[\overline{x}/\overline{a}]$ be the related formula obtained by assigning $\overline{a}$ to $\overline{x}$. Let $\Sigma^A, s \vdash \phi[\overline{x}/\overline{a}]$ denote that, according to the semantic model checker, $\phi[\overline{x}/\overline{a}]$ is* true *which refers to Σ^A and state s. Then, the outcome of the semantic model checker is defined as follows:*

$$\textsc{smc}(\Sigma^A, \phi[\overline{x}]) = \{\overline{a} \mid I(\overline{a}) \in (\Delta^{\mathcal{I}})^n \text{ and } \exists\, s \in \mathcal{S}^0, \Sigma^A, s \vdash \phi[\overline{x}/\overline{a}]\}$$

The fact that there exists at least an assignment (i.e. $\textsc{smc}(\Sigma^A, \phi[\overline{x}]) \neq \emptyset$) is denoted as: $\Sigma^A \vdash \phi$.

For the sake of space, here it is reported only a short introduction to such an algorithm. The reader may refer to [4] for a more detailed description and to Appendix A for the pseudocode of the semantic model checking algorithm. The steps of the algorithm are the following. First, a given temporal conjunctive query is translated into its positive normal form (*temporal conjunctive query normalization*), a more convenient form for the semantic model checking algorithm. The normal form can be obtained by pushing negations inward as far as possible. Then, *temporal knowledge base grounding* consists in applying to each ABox in Σ^A, each conjunctive query in ϕ. This step aims at mapping conjunctive queries in ϕ into propositions in order to obtain a finite STS. According to the closed world assumption, when a conjunctive query has a non-empty answer in a given state (ABox), the corresponding proposition is set to true in that state, to false otherwise. All the assignments, that represent an answer for any conjunctive query, are joined in a table called $\mathcal{A}$SSER in the algorithms in Appendix A. Regarding the query, *temporal conjunctive query grounding* is performed to obtain a set of propositional queries containing conditions on state propositions in place of queries of the original formula ϕ. Considering the closed world assumption, this step generates a set of propositional (ground) temporal specifications, one for each tuple in table $\mathcal{A}$SSER. At this point, the entailment problem has been reduced into a propositional model checking problem. Therefore, *propositional model checking* aims at verifying for each tuple in $\mathcal{A}$SSER whether the related propositional temporal specification is satisfied. If so, the tuple is inserted in $\textsc{smc}(\Sigma^A, \phi)$. At the end, $\textsc{smc}(\Sigma^A, \phi)$ contains all the answers to the temporal conjunctive query.

Now it is possible to state the following theorem about the soundness and completeness of the proposed approach.

Theorem 1 (Soundness & Completeness).

$$\Sigma^A, s \vdash \phi \;\; \textit{iff} \;\; \mathcal{K}, s \models \phi \tag{1}$$

$$\Sigma^A \vdash \phi \;\; \textit{iff} \;\; \mathcal{K} \models \phi \tag{2}$$

Proof. [Hint] The proof of (1) is reported in [4]. From (1) it follows that SMC$(\Sigma^A, \phi[\overline{x}]) = Ans(\mathcal{K}, \phi[\overline{x}])$. *Hence, it follows (2)*

Given that the entailment and answering problems (Definitions 6 and 7) are solved using semantic model checking, their complexities can be evaluated with the complexity of the semantic model checking algorithm. It follows:

Theorem 2 (Complexity). *The semantic model checking algorithm is decidable in time*

$$\max(\mathcal{O}(|\Sigma^A| \cdot |\phi| \cdot |CQA|)\,,\, \mathcal{O}(|\Sigma^A| \cdot |MC|))$$

where $|\Sigma^A|$ is proportional to the number and size of the states, $|\phi|$ is proportional to the length of ϕ, $|CQA|$ is the complexity of conjunctive query answering, and $|MC|$ is the complexity of propositional temporal logic model checking.

Proof. [Hint] The semantic model checking algorithm calls the conjunctive query answering algorithm $n\times \mid \mathcal{S} \mid$ times, where n is the number of conjunctive queries in the temporal specification ϕ (proportional to $|\phi|$) and $\mid \mathcal{S} \mid$ is the number of states (it is proportional to $|\Sigma^A|$). The result of conjunctive query answering is used to build a set of propositions whose number is less than or equal to $m\cdot \mid (\Delta^{\mathcal{I}})^h \mid$. The algorithm builds a join table in polynomial time. From this table, a set of propositional temporal specifications is built in polynomial time. Finally, the semantic model checking algorithm performs a traditional propositional temporal logic model checking a number of times that is proportional to the size of the join table. The size of the join table depends on the number of individuals in $\Delta^{\mathcal{I}}$ and, thus, on $|\Sigma^A|$.

6 Discussion and Conclusion

In this paper is presented a minimalistic approach to the problem of query answering in temporal knowledge bases.

Results reported in Section 5 show how temporal reasoning with a temporal knowledge base, under the *minimalistic approach* (i.e. the assumptions introduced in Section 3: DL-Lite$_{core}$, CTL, time invariant TBox, time variant ABox, constant and finite domain, closed-world assumption) is tractable. Up to now, such an approach has been confirmed to be useful to solve several verification (entailment) problems: semantic web service identification [27], selection [16], composition [15], and security [28]. Furthermore, its application to entailment problems ranges from business processes [29] to e-learning processes [30]. The original contribution of this paper consists in proposing to apply the *minimalistic approach* to query-answering problems as well, maintaining the same complexity and the same assumptions. As a matter of fact, it is still possible to use the same semantic model checking algorithm (as discussed in Section 5). Some practical experiments in real world applications of the proposed approach to temporal query-answering has been planned as future work.

References

1. S. Agarwal. A goal specification language for automated discovery and composition of web services. In *International Conference on Web Intelligence*, Silicon Valley, USA, 2007.
2. A. Deutsch, L. Sui, V. Vianu, and D. Zhou. Verification of communicating data-driven web services. In *PODS '06: Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 90–99, New York, NY, USA, 2006. ACM.
3. Victor Vianu. Automatic verification of database-driven systems: a new frontier. In *Proceedings of the 12th International Conference on Database Theory*, pages 1–13. ACM, 2009.
4. I. Di Pietro, F. Pagliarecci, and L. Spalazzi. Model checking semantically annotated services. *Software Engineering, IEEE Transactions on*, 38(3):592–608, 2012.
5. S. Hallé, R. Villemaire, and O. Cherkaoui. Specifying and validating data-aware temporal web service properties. *IEEE Trans. on Software Engineering*, 99(2):669–683, 2009.
6. Daniela Grigori, Juan Carlos Corrales, Mokrane Bouzeghoub, and Ahmed Gater. Ranking bpel processes for service discovery. *Services Computing, IEEE Transactions on*, 3(3):178–192, 2010.
7. Ahmed Awad, Artem Polyvyanyy, and Mathias Weske. Semantic querying of business process models. In *Enterprise Distributed Object Computing Conference, 2008. EDOC'08. 12th International IEEE*, pages 85–94. IEEE, 2008.
8. Babak Bagheri Hariri, Diego Calvanese, Giuseppe De Giacomo, Riccardo De Masellis, Paolo Felli, and Marco Montali. Verification of description logic knowledge and action bases. In *Proc. of the 20th Eur. Conf. on Artificial Intelligence (ECAI 2012)*, 2012.
9. Franz Baader, Stefan Borgwardt, and Marcel Lippmann. Temporalizing ontology-based data access. In *Proceedings of the 24th International Conference on Automated Deduction (CADE-24)*, Lecture Notes in Artificial Intelligence, Lake Placid, NY, USA, 2013. Springer-Verlag. To appear.
10. Feng Yang, JiuWei Wang, and Hemin Jin. A study on the customer behavior tracking model based on temporal description logic in the process of e-commerce. In *Systems and Informatics (ICSAI), 2012 International Conference on*, pages 2289–2293, 2012.
11. C. Lutz, F. Wolter, and M. Zakharyaschev. Temporal description logics: A survey. In S. Demri and C.S. Jensen, editors, *15th International Symposium on Temporal Representation and Reasoning, TIME 2008*, pages 3–14. IEEE Computer Society, 2008.
12. A. Artale and E. Franconi. A survey of temporal extensions of description logics. *Ann. Math. Artif. Intell.*, 30(1-4):171–210, 2000.
13. Klaus Schild. Combining terminological logics with tense logic. In Miguel Filgueiras and Lus Damas, editors, *Progress in Artificial Intelligence*, volume 727 of *Lecture Notes in Computer Science*, pages 105–120. Springer Berlin Heidelberg, 1993.
14. S. Agarwal. *Formal Description of Web Services for Expressive Matchmaking*. PhD thesis, Universität Karlsruhe (TH), Fakultät für Wirtschaftswissenschaften, 2007.
15. M. Pistore, L. Spalazzi, and P. Traverso. A minimalist approach to semantic annotations for web processes compositions. In *Proc. of the 3rd European Semantic Web Conference (ESWC 2006)*. Springer–Verlag, Berlin, Germany, 2006.

16. I. Di Pietro, F. Pagliarecci, L. Spalazzi, A. Marconi, and M. Pistore. Semantic Web Service Selection at the Process-Level: The eBay/Amazon/PayPal Case Study. In *Web Intelligence*, pages 605–611, 2008.
17. F. Weitl, M. Jakšić, and B. Freitag. Towards the automated verification of semi-structured documents. *Journal of Data & Knowledge Engineering*, 68:292–317, 2009.
18. F. Weitl, S. Nakajima, and B. Freitag. Structured counterexamples for the temporal description logic alcctl. In *Software Engineering and Formal Methods (SEFM), 2010 8th IEEE International Conference on*, pages 232–243, 2010.
19. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data Complexity of Query Answering in Description Logics. In *Tenth International Conference on Principles of Knowledge Representation and Reasoning*. AAAI Press, 2006.
20. Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The dl-lite family and relations. *Journal of Artificial Intelligence Research*, 36(1):1–69, 2009.
21. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. *Artificial Intelligence*, 2012. cited By (since 1996)1; Article in Press.
22. S. S. Lam and A. U. Shankar. A relational notation for state transition systems. *IEEE Trans. on Softw. Eng.*, 16(7):755, July 1990.
23. E. M. Clarke and J. M. Wing. Formal methods: State of the art and future directions. *ACM Computing Surveys*, 28(4):626–643, December 1996.
24. E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 14, pages 996–1072. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands, New York, N.Y., 1990.
25. E. Allen Emerson. Model checking and the mu-calculus. In Neil Immerman and Phokion G. Kolaitis, editors, *Descriptive Complexity and Finite Models, Proceedings of a DIMACS Workshop, January 14-17, 1996, Princeton University*, volume 31 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 185–214. American Mathematical Society, 1996.
26. M. Ortiz, D. Calvanese, and T. Eiter. Characterizing Data Complexity for Conjunctive Query Answering in Expressive Description Logics. In *21st AAAI Conference on Artificial Intelligence*, pages 275–280. AAAI Press, 2006.
27. D. Bianchini, F. Pagliarecci, and L. Spalazzi. From Service Identification to Service Selection: An Interleaved Perspective. In G. Agha, O. Danvy, and J. Meseguer, editors, *Formal Modeling: Actors, Open Systems, Biological Systems - Essays Dedicated to Carolyn Talcott on the Occasion of Her 70th Birthday*, volume 7000 of *Lecture Notes in Computer Science*, pages 223–240. Springer, 2011.
28. L. Boaro, E. Glorio, F. Pagliarecci, and L. Spalazzi. Model Checking Security Requirements for Web Services. In *International Conference on High Performance Computing Simulation (HPCS '10)*, june 2010.
29. L. Boaro, E. Glorio, F. Pagliarecci, and L. Spalazzi. Business process design framework for B2B collaboration. In W. Smari and G. Fox, editors, *2011 International Conference on Collaboration Technologies and Systems (CTS 2011)*, pages 633–635, Philadelphia, Pennsylvania, USA, May 23-27, 2011. IEEE.
30. T. Leo, F. Pagliarecci, and L. Spalazzi. eLearning for Complex System Professionals: Material Knowledge Representation, Retrieval, and Building. In *4th International Forum on Knowledge Asset Dynamics (IFKAD 09)*, Glasgow, United Kingdom, February 17–18 2009.

A Algorithms

```
Algorithm 1: semantic_model_checking
input  : $\Sigma^A = \langle\langle \mathcal{S}, \mathcal{S}^0, \mathcal{R}\rangle, \mathcal{T}, \Lambda\rangle$ /* ASTS */
         $\phi = \phi(q_1, \dots, q_m)$          /* Temporal conjunctive query */
output: $b \in \{true, false\}$             /* A boolean value */
         $\text{SMC}(\Sigma^A, \phi)$                /* The answer to the temporal conjunctive query */
{
    tcq_normalization(in: $\phi$, out: $\phi'$);
    tkb_grounding(in: $\Sigma^A, \phi'$, out: $\Sigma, \mathcal{A}\text{SSER}$);
    tcq_grounding(in: $\phi', \mathcal{A}\text{SSER}$, out: $\Phi$)
    $b := false$;
    $\text{SMC}(\Sigma^A, \phi) := \emptyset$
    for each $\varphi(\overline{r}) \in \Phi$ do
        {
            if prop_model_checking(in: $\Sigma, \varphi(\overline{r})$) then
                {
                    $b := true$
                    $\text{SMC}(\Sigma^A, \phi) := \text{SMC}(\Sigma^A, \phi) \cup \{\overline{r}\}$
                }
        }
    return $b$;
}
```

Fig. 1. The semantic model checking algorithm.

```
Algorithm 2: tkb_grounding
input  : $\Sigma^A = \langle\langle\mathcal{S}, \mathcal{S}^0, \mathcal{R}\rangle, \mathcal{T}, \Lambda\rangle$ /* ASTS */
         $\phi' = \phi'(q_1, \ldots, q_m)$        /* Temporal conjunctive query */
output: $\Sigma = \langle\langle\mathcal{S}, \mathcal{S}^0, \mathcal{R}\rangle, \mathcal{P}, \mathcal{X}\rangle$   /* Finite STS */
         $\mathcal{A}$SSER                          /* Join of all the tables */
{
    $\Delta := \emptyset$;
    $\mathcal{P} := \emptyset$
    for each $q_i$ $(1 \leq i \leq m)$ do  $\mathcal{A}sser(q_i) := \emptyset$;
    for each $s \in \mathcal{S}$ do {
        for each $a : C \in \Lambda(s)$ do  $\Delta := \Delta \cup \{a\}$;
        for each $a.R = b \in \Lambda(s)$ do  $\Delta := \Delta \cup \{a, b\}$;
    }
    for each $s \in \mathcal{S}$ do {
        $\mathcal{X}(s) := \emptyset$ ;
        for each $q_i$ $(1 \leq i \leq m)$ do {
            $I(q_i, s) :=$ cq_answering$(q_i, \langle\mathcal{T}, \Lambda(s)\rangle)$;
            if $\neg q_i \in \phi'$ then
                $I(q_i, s) := \Delta^h \setminus I(q_i, s)$;
            $\mathcal{A}sser(q_i) := \mathcal{A}sser(q_i) \cup I(q_i, s)$;
            for each $p_j(\overline{x}_j) \in q_i$ do
                for each $\overline{r} \in \pi_{\overline{x}_j}(I(q_i, s))$ do {
                    $\mathcal{P} := \mathcal{P} \cup \{PROP(p_j(\overline{r}))\}$;
                    if not $(\neg q_i \in \phi')$ then
                        $\mathcal{X}(s) := \mathcal{X}(s) \cup \{PROP(p_j(\overline{r}))\}$;
                    else if $\langle\mathcal{T}, \Lambda(s)\rangle \vdash p_j(\overline{r})$
                        $\mathcal{X}(s) := \mathcal{X}(s) \cup \{PROP(p_j(\overline{r}))\}$;
                }
        }
    }
    $\mathcal{A}$SSER $= \mathcal{A}sser(q_1) \bowtie \ldots \bowtie \mathcal{A}sser(q_m)$;
    return $\Sigma$ , $\mathcal{A}$SSER;
}
```

Fig. 2. The temporal ABoxes knowledge base grounding algorithm.

```
Algorithm 3: tcq_grounding
input  : $\phi'$       /* Temporal conjunctive query */
         $\mathcal{A}$SSER /* Join of all the tables */
output: $\Phi$       /* Set of ground CTL formulae */
{
    $\Phi := \emptyset$;
    for each $\overline{r} \in \mathcal{A}$SSER do {
        $\varphi(\overline{r}) := \phi'$;
        for each $x \in \mathsf{V}(\phi')$ do
            /* substitute each occurence of $x$ with $\pi_x(\overline{r})$ */
            $\varphi(\overline{r}) := \varphi[x/\pi_x(\overline{r})]$;
        $\Phi := \Phi \cup \{\varphi(\overline{r})\}$;
    }
    return $\Phi$
}
```

Fig. 3. The temporal conjunctive query grounding algorithm.