Cleaning Inconsistent Data in Temporal DL-Lite Under Best Repair Semantics

Sabiha Tahrat, Salima Benbernou, Mourad Ouzirri

Cleaning Inconsistent Data in Temporal *DL-Lite* Under Best Repair Semantics

Sabiha Tahrat
LIPADE, Université de Paris, France

Salima Benbernou
LIPADE, Université de Paris, France

Mourad Ouziri
LIPADE, Université de Paris, France

Abstract

In this paper, we address the problem of handling inconsistent data in Temporal Description Logic (TDL) knowledge bases. Considering the data part of the knowledge base as the source of inconsistency over time, we propose an ABox repair approach. This is the first work handling the repair in TDL Knowledge bases. To do so, our goal is twofold: 1) detect temporal inconsistencies and 2) propose a data temporal reparation. For the inconsistency detection, we propose a reduction approach from TDL to DL which allows to provide a tight NP-complete upper bound for TDL concept satisfiability and to use highly optimised DL reasoners that can bring precise explanation (the set of inconsistent data assertions). Thereafter, from the obtained explanation, we propose a method for automatically computing the best repair in the temporal setting based on the allowed rigid predicates and the time order of assertions.

2012 ACM Subject Classification Theory of computation → Description logics; Theory of computation → Modal and temporal logics

Keywords and phrases Inconsistency management, Temporal data, TDL, Data reparation.

Digital Object Identifier 10.4230/LIPIcs.TIME 2021.2021.

1 Introduction

The Ontology Web Language (OWL) is the ontology language recommended by the W3C that can be used to model knowledge domains. OWL is derived from the well known Description Logics (DLs) [6] which provide the basic representation features of OWL. Despite the expressiveness power of OWL, it cannot fully express the temporal knowledge needed in many applications. Beyond allowing data values to be typed as basic XML Schema dates, times or durations [1], OWL has a very limited support for temporal information modeling and reasoning. Time is crucial because events occur at specific points in time and also the relationships among objects exist over time. The ability to model this temporal dimension is therefore crucial in real- world applications such as banking, medical records and geographical information systems. Thus, the temporalization of DL (TDL) has been studied in [3, 20] but for reasoning tasks such as satisfiability, it is known to be hard. Temporal extensions of the lightweight DL «DL-Lite» [5, 12] are therefore considered for their low complexity for many reasoning problems. As *TDL-Lite* language we here consider the $T^{\mathbb{N}}$*DL-Lite* fragment [25] allowing for: full Boolean connectives, LTL operators [23] interpreted over $\mathbb{N}$ in the construction of concepts, inverse roles, distinction between local and global roles (aka. rigid roles). And we restrict ourselves to using the two future operators ($\Diamond_F$: eventually in the future and $\Box_F$: always in the future) and to specifying functionality on roles or their inverses.

[1] https://www.w3.org/TR/xmlschema11-1/

Note that, $T^{\mathbb{N}}$*DL-Lite* is the simpler logic of T_{FPX}*DL-Lite*$^{\mathcal{N}}_{bool}$fragment, the most expressive combination of the tractable *DL-Lite* family logics with LTL and which is known to be NP-*complete* [5].

TDL-Lite ontologies, also called *temporal knowledge bases* (TKBs), are expressed as a finite set of *general concept inclusions GCIs TBox*, which is expressed with temporalized concepts, paired with a timestamped factual knowledge *ABox* that represents data at different time points. Therefore, one of the most important challenges in TDL is to deal with inconsistent ontologies where the *ABox* is inconsistent with a satisfiable TBox: a subset of the assertions in the *ABox* contradicts one or more *TBox* assertions.

Then, the ABox is not reliable and must be repaired. The problem of handling inconsistent data in TKBs has not been fully addressed and only focused so far on satisfiability checking [25]. The reparation of the inconsistent data assertions in the ABox has not however been addressed yet. To the best of our knowledge, we propose the first approach to automatically repair *ABox* over TDL KBs based on the maximal repair semantic. The obtained repair is a maximal subset of the *ABox* that is consistent with the *TBox*. More precisely, we make the following contributions:

- We present a linear equisatisfiable translation of *TDL-Lite* knowledge bases KBs into DL KBs which allows to provide a tight NP-*complete* upper bound for *TDL-Lite* concept satisfiability and to use highly optimized DL reasoners that can bring precise inconsistency explanations (the set of inconsistent data assertions).
- We adapt the existing automatic repairing approach of an inconsistent DL ontology, based on the maximal repair semantic [8], for temporal *ABoxes* and we show that this semantic is preserved.
- We extend the maximal repair semantic in DL to the temporal setting by associating the detected inconsistent assertions with a temporal weight. This weighting is based on the defined rigid predicates in the *TBox* and the timestamp order of the *ABox* assertions. The repair is computed by removing inconsistent assertions with the lowest weight.

In the following, we present our running example to provide an intuitive overview of the inconsistency detection and the repair over *TDL-Lite* KBs.

▶ **Example 1.** Let's consider the following KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ where $\mathcal{T}$ is a *TDL-Lite* TBox stating that minors and adults are persons, but they are disjoint. Moreover, the concepts person, adult and the role *hasMother* are rigid. The ABox $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ reports *John's* status in $\mathcal{A}_1$ and his mother in $\mathcal{A}_2$, at different timestamps:

$$\begin{aligned}
\mathcal{T} = \quad & \{\mathsf{Adult} \sqsubseteq \mathsf{Person},\ \mathsf{Minor} \sqsubseteq \mathsf{Person},\ \mathsf{Person} \sqsubseteq \Box_F \mathsf{Person},\ \mathsf{Adult} \sqsubseteq \Box_F \mathsf{Adult} \\
& \mathsf{Minor} \sqcap \mathsf{Adult} \sqsubseteq \bot,\ \mathsf{Person} \sqsubseteq \exists \mathsf{hasMother},\ \ \mathsf{Funct}(\mathsf{hasMother})\} \\
\mathcal{A}_1 = \quad & \{\mathsf{Person}(\mathsf{John}, 0), \mathsf{Minor}(\mathsf{John}, 1), \mathsf{Adult}(\mathsf{John}, 2), \mathsf{Minor}(\mathsf{John}, 3), \mathsf{Minor}(\mathsf{John}, 4)\} \\
\mathcal{A}_2 = \quad & \{\mathsf{hasMother}(\mathsf{John}, \mathsf{Ana}, 0), \mathsf{hasMother}(\mathsf{John}, \mathsf{Eva}, 1), \mathsf{hasMother}(\mathsf{John}, \mathsf{Maria}, 2)\}
\end{aligned}$$

Both $\mathcal{A}_1$ and $\mathcal{A}_2$ are inconsistent w.r.t $\mathcal{T}$. In $\mathcal{A}_1$, the assertions: $\mathsf{Adult}(\mathsf{John}, 2)$, $\mathsf{Minor}(\mathsf{John}, 3)$ and $\mathsf{Minor}(\mathsf{John}, 4)$ with the rigidity of adult violate the disjointness between adult and minor. Similarly in $\mathcal{A}_2$, John has multiple mothers at different time points which violates the properties (functional and global) of the role $\mathsf{hasMother}$. Then $\mathcal{A}_1$ has a maximal repair $\mathcal{A}'_1 = \{\mathsf{Person}(\mathsf{John}, 0), \mathsf{Minor}(\mathsf{John}, 1), \mathsf{Minor}(\mathsf{John}, 3), \mathsf{Minor}(\mathsf{John}, 4)\}$ and $\mathcal{A}_2$ has three possible maximal repairs as follows:

$$\mathcal{A}'_2 = \{\mathsf{hasMother}(\mathsf{John}, \mathsf{Ana}, 0)\} \qquad \mathcal{A}''_2 = \{\mathsf{hasMother}(\mathsf{John}, \mathsf{Eva}, 1)\}$$
$$\mathcal{A}'''_2 = \{\mathsf{hasMother}(\mathsf{John}, \mathsf{Maria}, 2)\}$$

This paper is structured as follows. In the next section, we sketch works that have been conducted in the context of inconsistent data reparation in knowledge base fields. In section

3, we introduce the syntax and semantics of the temporal DL *TDL-Lite* that formalise our running example. In section 4, we propose a translation to reduce TDL KBs to DL KBs. Based on the obtained DL KBs, in section 5, we perform inconsistency detection and compute the best temporal data repair. Section 6 concludes this paper and presents some future works.

2 Related Work

The problem of inconsistencies appearing in KBs can be tackled either by repairing the KB, which leads to a consistent version of it [15], or by providing the ability to query inconsistent data and get consistent answers (Consistent Query Answering - CQA) [2]. These two approaches were applied initially in the context of relational databases and, later, in the context of KBs as well. The repair approaches share the same principle of performing a minimal set of actions (insertions, deletions, updates) over the KB, in order to render it valid with respect to a given set of integrity constraints. They differ in the type of integrity constraints and in the applied actions. However, CQA approaches take into account, at query execution time, all the possible repairs without materializing a repair of the inconsistent KB. They differ in the repair semantics, the type of integrity constraints and ways of computing the repairs. In the following, we will refer to the most notable related works, dealing with the problems of CQA or repairing, that have been proposed for atemporal and temporal KBs.

2.1 Atemporal KBs Repair

The most well-known, and arguably the most natural approach for Consistent Query Answering is the ABox Repair (AR) semantics [18]. It consists of finding all the maximal subsets of the ABox that are consistent with the TBox and thus showing that inconsistency-tolerant instance checking is already intractable. For this reason, the Intersection ABox Repair (IAR) semantics was proposed which is the intersection of all AR-repairs, and is polynomially tractable. Few implemented systems [10] designing practically efficient consistent query answering systems, that could scale up to billions of data, is still largely open. There has been few implementations regarding ABox inconsistency checking. The reasoner QuOnto [1] allows to only check the satisfiability of *DL-Lite*$_A$ KBs the simplest fragment of *DL-Lite* which is at the bases of OWL 2 QL [2]. A preliminary ABox cleaner system QuAC implemented within QuOnto is reported by [21] over *DL-Lite*$_A$ KBs. It is based on the semantics discussed in [19] where each inconsistency is resolved by removing all data assertions that take part in it and the evaluation was conducted for datasets of few thousand assertions. On the contrary, in [8] the repair is processed on big RDF KBs by only removing one triple from the interdependent inconsistent triples.

2.2 Temporal KBs Repair

Very few works have investigated the repairing of inconsistent temporal knowledge base. So far, query answering has been extended to the temporal setting in light-weight DLs over inconsistent data, allowing both rigid concepts and roles whose interpretations do not change over time and different types of repair semantics [11]. In this work, temporal operators are only used in the definition of queries which are applied to *static* ontologies together with

[2] https://www.w3.org/TR/owl2-profiles/

sequences of datasets at time points. The ontology along with the sequence of datasets constitute the temporal knowledge base. Unlike the quoted work, we here use *dynamic* ontologies that allow temporal operators in the definition of concepts. A recent work tackled this issue by translating TDL-Lite KBs into LTL formulas where LTL reasoners can be applied to check their satisfiability [25]. However, root causes of unsatisfiability, when it exits, are not identified. In this paper, we extend existing approaches of detecting the minimal inconsistent subset in Description Logic (DL) knowledge bases to the temporal setting. We make the assumption that ABox assertions are reliable over time and we focus on repairing the data in the ABox based on the TBox specifications and under the best temporal repair semantic.

3 Temporal Description Logic

We now provide details about the syntax and the semantics of the temporal description logic *TDL-Lite*. In our study as *TDL-Lite* we consider the $T^{\mathbb{N}}$*DL-Lite* fragment, that allows only future operators interpreted over $\mathbb{N}$ to concepts. We further impose the only use of the two future temporal operators: $\Diamond_F$(eventually in the future) and $\Box_F$(always in the future) and applied only in the right-hand side of inclusions.

▶ **Definition 2.** *Let* $\mathsf{N_C}, \mathsf{N_I}$ *and* $\mathsf{N_R}$ *be countable sets of* concept, individual names *and* roles *respectively.* $\mathsf{N_R}$ *is the union* $\mathsf{N_G} \cup \mathsf{N_L}$ *where* $\mathsf{N_G}$ *and* $\mathsf{N_L}$ *are countable and disjoint sets of* global *and* local role names*, respectively. TDL-Lite* basic concepts B, concepts C,(temporal) concepts D*, and* roles R*, are formed according to the following grammar:*

$$\begin{array}{llll} R & ::= L \mid L^- \mid G \mid G^-, & B & ::= \bot \mid \top \mid A \mid, \exists R, \\ C & ::= B \mid \neg C \mid C_1 \sqcap C_2, & D & ::= C \mid \Diamond_F D \mid \Box_F D \mid \neg D \mid D_1 \sqcap D_2. \end{array}$$

where $L \in \mathsf{N_L}$, $G \in \mathsf{N_G}$, $A \in \mathsf{N_C}$*. we called* disjointness*, inclusions of the form* $C \sqcap D \sqsubseteq \bot$*. We also add the ability to specify* functional *roles (funct* R*).*

A *TDL-Lite knowledge base*, $\mathcal{K}$, is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox. A *TBox* is a finite set of *general concept inclusions (GCI)* of the form $C \sqsubseteq D$ where C, D are *TDL-Lite* concepts, and an *ABox* is a finite set of *concept assertion* of the form $\bigcirc^n A(a)$ or $\bigcirc^n \neg A(a)$, or a *role assertion* of the form $\bigcirc^n R(a, b)$ or $\bigcirc^n \neg R(a, b)$, $A \in \mathsf{N_C}$, $R \in \mathsf{N_R}$, $a, b \in \mathsf{N_I}$, and $n \in \mathbb{N}$. Here we assume that there is no time point before 0 or after n and we use abbreviations for assertions $A(a, n) = \bigcirc^n A(a)$ and $R(a, b, n) = \bigcirc^n R(a, b)$.

▶ **Definition 3.** *A TDL-Lite* interpretation *is a structure* $\mathfrak{M} = (\Delta^{\mathfrak{M}}, (\mathcal{I}_n)_{n\in\mathbb{N}})$*, where each* $\mathcal{I}_n$ *is a classical DL interpretation with non-empty domain* $\Delta^{\mathfrak{M}}$ *(or simply* Δ*). We have that* $A^{\mathcal{I}_n} \subseteq \Delta^{\mathfrak{M}}$ *and* $S^{\mathcal{I}_n} \subseteq \Delta^{\mathfrak{M}} \times \Delta^{\mathfrak{M}}$*, for all* $A \in \mathsf{N_C}$ *and* $S \in \mathsf{N_R}$*. In particular, Rigid predicates are elements from the set of rigid concepts* $\mathsf{N_{RC}} \subseteq \mathsf{N_C}$ *or of rigid roles* $\mathsf{N_G}$ *and for all* $X \in \mathsf{N_{RC}} \cup \mathsf{N_G}$ *and* $i, j \in \mathbb{N}$, $X^{\mathcal{I}_i} = X^{\mathcal{I}_j}$ *(denoted simply by* $X^{\mathcal{I}}$*). Moreover,* $a^{\mathcal{I}_i} = a^{\mathcal{I}_j} \in \Delta^{\mathfrak{M}}$ *for all* $a \in \mathsf{N_I}$ *and* $i, j \in \mathbb{N}$*, i.e., constants are* rigid designators *(with fixed interpretation, denoted simply by* $a^{\mathcal{I}}$*). We assume that all interpretations* $\mathfrak{M}$ *satisfy the unique name assumption UNA and the* constant domain assumption *(meaning that objects are not created nor destroyed over time). The interpretation of roles and concepts at instant* $n \in \mathbb{N}$ *is defined as follows (where* $S \in \mathsf{N_R}$*):*

$$\begin{array}{llll} (S^-)^{\mathcal{I}_n} & = \{(d', d) \in \Delta^{\mathfrak{M}} \times \Delta^{\mathfrak{M}} \mid (d, d') \in S^{\mathcal{I}_n}\}, & \bot^{\mathcal{I}_n} & = \emptyset, \\ (\exists R.D)^{\mathcal{I}_n} & = \{d \in \Delta^{\mathfrak{M}} \mid \exists d' \in D^{\mathcal{I}_n} : (d, d') \in R^{\mathcal{I}_n}\}, & (\neg D)^{\mathcal{I}_n} & = \Delta^{\mathfrak{M}} \setminus D^{\mathcal{I}_n}, \\ (D_1 \sqcap D_2)^{\mathcal{I}_n} & = D_1^{\mathcal{I}_n} \cap D_2^{\mathcal{I}_n}, & (D_1 \sqcup D_2)^{\mathcal{I}_n} & = D_1^{\mathcal{I}_n} \cup D_2^{\mathcal{I}_n}, \\ (\Diamond_F D)^{\mathcal{I}_n} & = \bigcup_{k>n} D^{\mathcal{I}_k}, & (\Box_F D)^{\mathcal{I}_n} & = \bigcap_{k>n} D^{\mathcal{I}_k}, \end{array}$$

▶ **Definition 4.** *We say that a concept D is* satisfied *in $\mathfrak{M}$ if there is $n \in \mathbb{N}$ such that $D^{\mathcal{I}_n} \neq \emptyset$. The* satisfaction of an axiom *in $\mathfrak{M}$ is defined as follows:*

$$\mathfrak{M} \models D_1 \sqsubseteq D_2 \quad \textit{iff} \quad D_1^{\mathcal{I}_n} \subseteq D_2^{\mathcal{I}_n} \textit{ for all } n \in \mathbb{N},$$
$$\mathfrak{M} \models \bigcirc^n A(a) \quad \textit{iff} \quad a^{\mathcal{I}} \in A^{\mathcal{I}_n}, \qquad \mathfrak{M} \models \bigcirc^n R(a,b) \quad \textit{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}_n}.$$
$$\mathfrak{M} \models \bigcirc^n \neg A(a) \quad \textit{iff} \quad a^{\mathcal{I}} \notin A^{\mathcal{I}_n}, \qquad \mathfrak{M} \models \bigcirc^n \neg R(a,b) \quad \textit{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \notin R^{\mathcal{I}_n}.$$

A KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is satisfiable *if it exists a model $\mathfrak{M}$ that satisfies every axiom of $\mathcal{T}$ and $\mathcal{A}$, and written $\mathfrak{M} \models \mathcal{K}$.*

▶ **Definition 5.** *An ABox $\mathcal{A}$ is $\mathcal{T}$-consistent if the KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is* satisfiable.

4 Reducing Temporal *DL-Lite* to *DL-Lite*

This section contains the reduction of *TDL-Lite* KBs, into *DL-Lite* KBs. This allows to provide a tight NP-*complete* upper bound for *TDL-Lite* concept satisfiability checking (as shown in section 4.1) and to use highly optimized DL reasoners that can identify the precise set of inconsistent data assertions. The translation is applied on both TBox and ABox levels in sections 4.2 and 4.3 respectively.

4.1 The Upper Bound

The temporal component of our $T^{\mathbb{N}}$*DL-Lite* fragment, as described in section 3, is based on the propositional LTL, more particularly LTL(F,G) using only the two modal operators $\Diamond_F \equiv F$: sometime in the future and $\Box_F \equiv G$: always in the future. The main idea is to consider two separate satisfiability problems, one in LTL and the other in *DL-Lite*, that together imply satisfiability of $\mathcal{K}$ in $T^{\mathbb{N}}$*DL-Lite*.

For the LTL part, it was shown in [14] that LTL(F,G) is the modal logic S4.3.1 (also called S4.3.Dum or D). Moreover, according to [22], the satisfiability of a S4.3.1 formula ϕ with a maximum nesting depth of modal operators equal to m (maximum temporal depth) is NP-*complete* and there exists a S4.3.1-model which satisfies ϕ with at most $m+1$ worlds. Furthermore, there exist an algorithm of a polynomial time complexity for transforming the LTL(F,G)-SAT problem into the SAT problem. As a consequence, a $T^{\mathbb{N}}$*DL-Lite* satisfiability problem, being NP-*complete*, can be reduced into a *DL-Lite* satisfiability problem since $m+1$ is an upper bound of the number of worlds in the model. The reduction is consequently equi-satisfiable to the original $T^{\mathbb{N}}$*DL-Lite* language. This is confirmed by the case of the upper bound shown by Ladner [22].

4.2 TBox Reduction to *DL-Lite*

Based on the temporal interpretation $(\mathcal{I}_n)_{n\in\mathbb{N}}$ of a TDL KB which is a standard DL interpretation $\mathcal{I}$ for each time instant (world) $n \in \mathbb{N}$ as described in definitions 3 and 4, we define the translation in the same way in the interval [0,m]. Given *TDL-Lite* concepts C, D and a role R, we inductively define the *DL-Lite* translation of concepts C, D and a role R at time point $i \in [0, m]$ denoted by tr(C,i,m), tr(D,i,m) and tr(R,i,m) as:

$$\begin{array}{llll}
tr(\top, i, m) & = \top, & tr(\bot, i, m) & = \bot, \\
tr(C, i, m) & = C_i, & tr(\neg C, i, m) & = \neg tr(C, i, m), \\
tr(C \sqcap D, i, m) & = tr(C, i, m) \sqcap tr(D, i, m), & tr(C \sqcup D, i, m) & = tr(C, i, m) \sqcup tr(D, i, m), \\
tr(\Box_F D, i, m) & = \sqcap_i^m \, tr(D, i, m), & tr(\Diamond_F D, i, m) & = \sqcup_i^m tr(D, i, m), \\
tr(R, i, m) & = R_i, & tr(\exists R, i, m) & = (\exists R_i).
\end{array}$$

The translation creates fresh concepts C_i, D_i and roles R_i in the *DL-Lite* TBox denoting respectively the interpretation $\mathcal{I}_i$ of C, D and R at time point i. Now, the translation $\mathcal{T}^\dagger$ of a TBox $\mathcal{T}$ is the conjunction of:

$$\bigwedge_{C \sqsubseteq D \in \mathcal{T}} \bigwedge_{i=0}^{m} (tr(C,i,m) \sqsubseteq tr(D,i,m)), \tag{1}$$

$$\bigwedge_{R \in \mathsf{N_G}} \bigwedge_{i=0}^{m} \big(tr(R,i,m) \sqsubseteq \sqcap_i^m \ \ tr(R,i,m)\big), \tag{2}$$

$$\bigwedge_{Funct(R)} \bigwedge_{i=0}^{m} (Funct(R_i)). \tag{3}$$

Note that due the translation of rigid roles (2), the resulting $\mathcal{T}^\dagger$ includes role inclusions. We restrict rigid roles to functional roles in this paper. This extension of *DL-Lite* with role inclusions and functional roles is denoted by *DL-Lite*$_{bool}^{(\mathcal{NH})}$. Its satisfiability problem is NP and matches that of the language without role inclusions. However, Local roles are not translated to role inclusions because their semantic is maintained in the resulting *DL-Lite* language.

▶ **Example 6.** We show the translation of the TBox $\mathcal{T}$ of example 1 where the maximum temporal depth over $\mathcal{T}$ formulas is $m = 1$. The resulting *DL-Lite* $\mathcal{T}^\dagger$ is:

$$\begin{aligned}\mathcal{T}^\dagger = \ & \{\mathsf{Adult}_0 \sqsubseteq \mathsf{Person}_0, \mathsf{Adult}_1 \sqsubseteq \mathsf{Person}_1, \mathsf{Minor}_0 \sqsubseteq \mathsf{Person}_0, \mathsf{Minor}_1 \sqsubseteq \mathsf{Person}_1 \\ & \mathsf{Person}_0 \sqsubseteq \mathsf{Person}_0 \sqcap \mathsf{Person}_1, \ \mathsf{Adult}_0 \sqsubseteq \mathsf{Adult}_0 \sqcap \mathsf{Adult}_1, \\ & \mathsf{Minor}_0 \sqcap \mathsf{Adult}_0 \sqsubseteq \bot, \mathsf{Minor}_1 \sqcap \mathsf{Adult}_1 \sqsubseteq \bot, \\ & \mathsf{Person}_0 \sqsubseteq \exists \mathsf{hasMother}_0, \ \mathsf{Person}_1 \sqsubseteq \exists \mathsf{hasMother}_1, \\ & \mathsf{Funct}(\mathsf{hasMother}_0), \ \mathsf{Funct}(\mathsf{hasMother}_1)\}\end{aligned}$$

4.3 ABox Reduction to *DL-Lite*

Now, we explain how an ABox $\mathcal{A}$ is translated to $\mathcal{A}^\dagger$. For each $n \in \mathbb{N}$, each concept $A, B \in \mathsf{N_C}$ and each role $R, S \in \mathsf{N_R}$, we define:

$$\mathcal{A}^\dagger = \bigwedge_{\bigcirc^n A(a) \in \mathcal{A}} A_n(a) \wedge \bigwedge_{\bigcirc^n R(a,b) \in \mathcal{A}} R_n(a,b), \bigwedge_{\bigcirc^n \neg B(a) \in \mathcal{A}} \neg B_n(a) \wedge \bigwedge_{\bigcirc^n \neg S(a,b) \in \mathcal{A}} \neg S_n(a,b), \tag{4}$$

$\mathcal{A}^\dagger$ is composed of four conjuncts, the first is the conjunction of the translation of all concept assertions in $\mathcal{A}$ and the second is the conjunction of the translation of all role assertions (global and local) occurring in $\mathcal{A}$. The last two conjunctions are equivalent to the first two conjunctions respectively when assertions are negated .

▶ **Example 7.** We show the translation of the ABox $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ in example 1. The resulting *DL-Lite* $\mathcal{A}^\dagger = \mathcal{A}_1^\dagger \cup \mathcal{A}_2^\dagger$ is:

$$\mathcal{A}_1^\dagger = \{\mathsf{Person}_0(\mathsf{John}), \mathsf{Minor}_1(\mathsf{John}), \mathsf{Adult}_2(\mathsf{John}), \mathsf{Minor}_3(\mathsf{John}), \mathsf{Minor}_4(\mathsf{John})\}$$

$$\mathcal{A}_2^\dagger = \{\mathsf{hasMother}_0(\mathsf{John}, \mathsf{Ana}), \mathsf{hasMother}_1(\mathsf{John}, \mathsf{Eva}), \mathsf{hasMother}_2(\mathsf{John}, \mathsf{Maria})\}$$

It is immediate to verify the satisfiability of the resulting $\mathcal{K}^\dagger = (\mathcal{T}^\dagger, \mathcal{A}^\dagger)$. However, concepts Adult_2, Minor_3, Minor_4 and $\mathsf{hasMother}_2$ created in $\mathcal{A}^\dagger$ do not occur in $\mathcal{T}^\dagger$ as computed in example 6. To overcome the above problem, the translated $\mathcal{T}^\dagger$ in the presence of an ABox $\mathcal{A}$ which is defined over a time interval $[l, n]$ should be computed in the interval $[l, n + m]$. The translation of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ into $\mathcal{K}^\dagger = \mathcal{T}^\dagger \wedge \mathcal{A}^\dagger$ is computed in polynomial time.

▶ **Example 8.** Giving the ABox $\mathcal{A}$ defined over the interval [1,4], the translation of the *TDL-Lite* TBox $\mathcal{T}$ in example 1 into *DL-Lite* is computed over the interval [1,5]:

$$\begin{aligned}\mathcal{T}^{\dagger} = \quad & \{\bigwedge_{i=1}^{5} \mathsf{Adult}_i \sqsubseteq \mathsf{Person}_i,\ \ \bigwedge_{i=1}^{5} \mathsf{Minor}_i \sqsubseteq \mathsf{Person}_i,\ \ \bigwedge_{i=1}^{5} \mathsf{Minor}_i \sqcap \mathsf{Adult}_i \sqsubseteq \bot,\\ & \bigwedge_{i=1}^{5} \mathsf{Person}_i \sqsubseteq (\sqcap_i^5 \mathsf{Person}_i),\ \bigwedge_{i=1}^{5} \mathsf{Adult}_i \sqsubseteq (\sqcap_i^5 \mathsf{Adult}_i),\\ & \bigwedge_{i=1}^{5} \mathsf{Person}_i \sqsubseteq \exists \mathsf{hasMother}_i,\ \ \bigwedge_{i=1}^{5} (\mathsf{hasMother}_i \sqsubseteq \sqcap_i^5 \mathsf{hasMother}_i),\\ & \bigwedge_{i=1}^{5} \mathsf{Funct}(\mathsf{hasMother}_i)\}\end{aligned}$$

▶ **Theorem 9.** *A $\mathbb{T}^{\mathbb{N}}$DL-Lite KB $\mathcal{K}$ is satisfiable iff the DL-Lite$_{bool}^{(\mathcal{NH})}$-formula $\mathcal{K}^{\dagger}$ is satisfiable. Moreover, $\mathcal{K}^{\dagger}$ can be constructed in polynomial time w.r.t. the size of $\mathcal{K}$.*

Proof. Theorem 9 is proved in the same way as the standard translation to FOL plus Theorem 6 in [22] . ◀

5 Inconsistency Detection and Repair in TKBs

Once *TDL-Lite* KBs are mapped into DL KBs as defined in the previous section, the following step is to detect the inconsistent data assertions using DL reasoners. In this section, we first recall the essentials of the DL inconsistency detection and then present the core procedure of computing the minimal inconsistent subset and the best repair in *TDL-Lite*.

5.1 Inconsistency Detection in KBs

The Web Ontology Language OWL is a logic-based language of knowledge representation intended to be used to verify the consistency of a dataset with the semantics of the underlying knowledge representation formalism. This task is usually handled by what we call OWL reasoners such as RacerPro[3], Pellet[4], FaCT++[5] and others. In FaCT++, nominals are unavailable and ABox reasoning is not supported. Nominals are also unvailable in RacerPro. Therefore, we use Pellet as our OWL reasoner: it is the first sound and complete tableau-based reasoner for the OWL-DL sublanguage (a syntactic variant of the Description Logic $\mathcal{SHOIN}(\mathcal{D})$ [24] which is much more expressive than the tractable *DL-Lite*$_{bool}^{(\mathcal{NH})}$). Pellet is written in Java, open source and efficient when the number of instances is large. Moreover, It also offers a specific service for computing inconsistency explanations on the TBox terminology and the assertional ABox levels. Moreover, Pellet outperforms RacerPro when reasoning on a large number of instances [24].

Before describing the inconsistency detection approach, we should explain what "inconsistent data assertion" means in the context of *DL-Lite*$_{bool}^{(\mathcal{NH})}$. We can distinguish three different types of *DL-Lite*$_{bool}^{(\mathcal{NH})}$ TBox constraints: disjointness which are GCIs of the form $(C \sqcap D \sqsubseteq \bot)$, functionality assertions of the form (funct R) and rigid predicates.

▶ **Definition 10.** *Let $\mathcal{K}^{\dagger} = (\mathcal{T}^{\dagger}, \mathcal{A}^{\dagger})$ be a DL-Lite$_{bool}^{(\mathcal{NH})}$ KB and let c be a disjointness inclusion or a functionality assertion or a rigid predicate of $\mathcal{T}^{\dagger}$. A set of data assertions $I = \langle a_1, a_2, ...\rangle \in \mathcal{A}^{\dagger}$, is called inconsistent, iff there is some $c \in \mathcal{T}^{\dagger}$, such that $I \not\models c$.*

[3] https://github.com/ha-mo-we/Racer
[4] http://pellet.owldl.com/
[5] http://owl.cs.manchester.ac.uk/tools/fact/

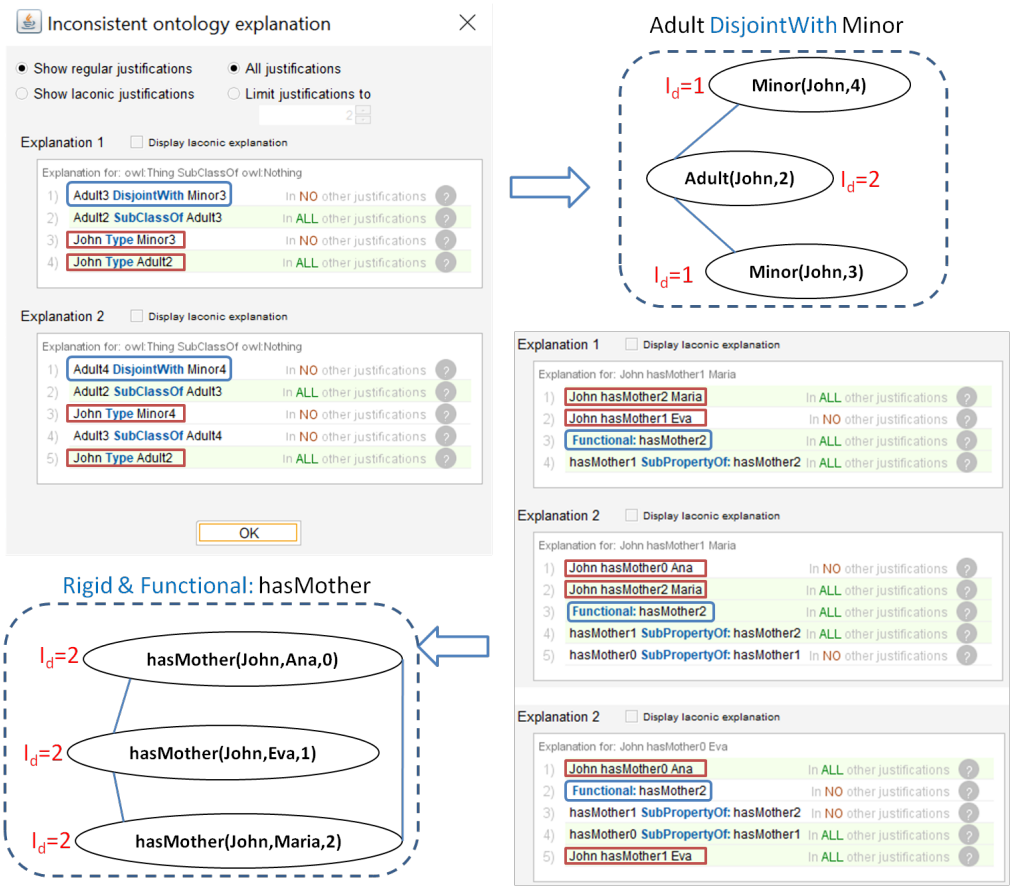


Figure 1 Inconsistency detection and explanation over example 1 giving by Pellet via the Protegé [6] Ontology editor. We give the corresponding Inconsistency Graph to show conflicts between the inconsistent assertions. We report the number of conflicts (number of edges) of each inconsistent assertion in the Inconsistency degree I_d.

Our approach starts by checking the satisfiability of the translated TBox $\mathcal{T}^\dagger$ then checks the consistency of the translated ABox $\mathcal{A}^\dagger$ according to $\mathcal{T}^\dagger$ using Pellet. If $\mathcal{A}^\dagger$ is inconsistent, the explanation support of Pellet points the inconsistent set of data assertions without a resolution strategy. However, this explanation support is an axiom tracing service which allows to extract, from the TBox and the ABox, the relevant axioms responsible for the inconsistency which can directly be used for reparation.

Based on the resulting *DL-Lite*$_{bool}^{(\mathcal{NH})}$ KB $\mathcal{K}^\dagger$ in section 4.3 of the motivating example, all the inconsistency explanations provided by Pellet are shown in figure 1. First, we can observe inconsistency due to conflicts in axioms: (John Type Adult$_2$) with (John Type Minor$_3$) and again (John Type Adult$_2$) with (John Type Minor$_4$) according to the disjointness CIs (Adult $\sqcap$ Minor $\sqsubseteq \bot$) translated to (Adult$_3$ DisjointWith Minor$_3$) and to (Adult$_4$ DisjointWith Minor$_4$) respectively at time point 3 in explanation 1 and at time point 4 in explanation 2. We also notice that axioms can be responsible for multiple conflicts according to the constraints in $\mathcal{T}^\dagger$. The axiom (JohnType Adult$_2$) which corresponds to the assertion Adult(John,2) in $\mathcal{A}$ has two conflicts according to explanations 1 and 2 in the left of figure 1. Formally, we define an *Inconsistency degree* I_d as the number of $\mathcal{T}^\dagger$ constraints (conflicts) on which an assertion is involved. Second, considering the rigid role HasMother, we can observe from the Pellet explanation, on the

right-side of figure 1, inconsistencies due to conflicts between axioms: (John hasMother_0 Ana), (John hasMother_1 Eva) and (John hasMother_2 Maria) according to the functional and rigid role hasMother. Consequently, each of these assertions has an $I_d = 2$. To report these inconsistent assertions in an intuitive way, we use an *Inconsistency graph* which is similar to the *conflict-hypergraph* used to represent constraint violations in databases [13] or in inconsistency DL setting [9, 8].

▶ **Definition 11.** *Let $\mathcal{K}^\dagger = (\mathcal{T}^\dagger, \mathcal{A}^\dagger)$ be a DL-Lite$_{bool}^{(\mathcal{NH})}$ KB and let $I_m = \langle a_i, a_j, ...\rangle \not\models c_k$ be sets of inconsistent data assertion where $c_k \in \mathcal{T}^\dagger$-constraints. An Inconsistency Graph of $\mathcal{K}^\dagger$ is an edge-labeled graph denoted by $IG(K) = (V, E)$ such that for all I_m, $V = \{a_i | a_i \in I_m\}$ and $E = \{(a_i, a_j) | \langle a_i, a_j\rangle \not\models c_k, c_k \in \mathcal{T}^\dagger\}$*

This graph is built by iterating over all Pellet explanations I_m as follows:

- for every assertion a_i of the form (a Type D_i) or (a R_i b), we add the vertice (D(a,i)) or (R(a,b,i));
- for every pair of vertices in I_m , we add an edge connecting them which we label with the broken constraint.

The inconsistency graph of the motivated example is shown in figure 1. Note that we can easily determine how many conflicts in which each inconsistent assertion is involved. This graph is used as an input for the repair phase which provides graph theories and tools useful for repair purposes.

5.2 Best Repair in KBs

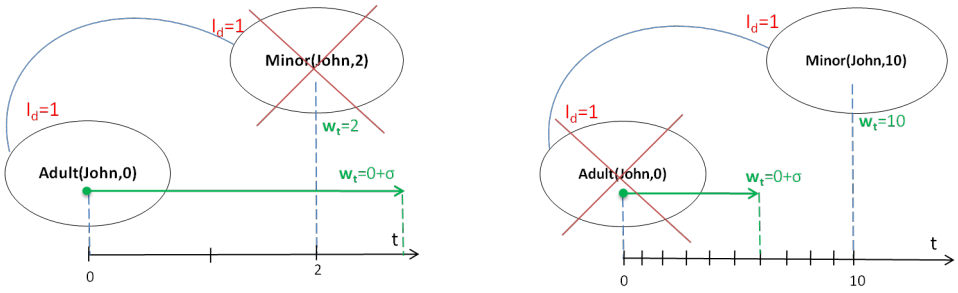
After building the inconsistency graph which encodes the inconsistent data assertions, the next step is to repair them. An extreme solution would be to simply throw away all the detected inconsistent assertions from $\mathcal{A}$. This would certainly not meet the expected repair requirement which consists of applying a minimal set of changes that restore consistency. Typically, minimality is defined by a set of inclusion yielding:

▶ **Definition 12.** *Consider a $T^{\mathbb{N}}$DL-Lite KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. A temporal ABox repair of $\mathcal{T}$ is a set $\mathcal{A}'$ of assertions such that: (i) $\mathcal{A}' \subseteq \mathcal{A}$; (ii) $\mathcal{K} = (\mathcal{T}, \mathcal{A}')$ is consistent; (iii) $\mathcal{A}' \subset \mathcal{A}'' \subseteq \mathcal{A}$ where $\mathcal{K} = (\mathcal{T}, \mathcal{A}'')$ is consistent and $\mathcal{A}''$ does not exist.*

In other words, $\mathcal{A}'$ is a maximal consistent subset of $\mathcal{A}$ that is obtained by throwing away a minimal set of inconsistent assertions (aka the Minimal Unsatisfiable Set MUS). This is performed by the removal of one of the two inconsistent data assertions involved in the conflict. In the inconsistency graph, this corresponds to removing one of the two vertices that are connected by the edge representing this conflict. A complete repair is in fact the well-known problem of finding the minimum vertex cover [17] which computes a set of vertices (a MUS) whose removal leads to the removal of all the edges (all the conflicts) of the inconsistency graph.

Recall that the computation of the minimum vertex cover is a classical NP-*complete* problem. However, an approximation algorithm, such as the 2-approximation algorithm in [17] can be applied in a greedy manner until there are no more edges in each connected component of the inconsistency graph as follows:

- for each step we select the vertex cover which is the vertex having a higher inconsistency degree I_d (the most inconsistent assertions are those involved in most conflicts) ;
- If more than one vertice have the same degree, one is randomly selected as vertex cover.



(a) case where the recent assertion is removed

(b) case where the rigid assertion is removed

Figure 2 Best temporal repair in TKBs

The union of vertex cover sets of the connected components of the graph forms a vertex cover (a MUS) of the entire inconsistency graph.

For instance, let's consider the inconsistency graph of figure 1. The minimum vertex cover algorithm will compute the repairs in the first connected component of the graph labeled by the constraint (Adult DisjointWith Minor) by removing the assertion Adult(John,2) because it has the highest I_d. In the second connected component labeled by the constraint (Rigid&Fuctional : hasMother), the repair will be computed by removing randomly two of the three assertions in the graph as they have the same I_d.

Let us note that there might be several possible MUSs for the same inconsistency graph, since some vertices can have the same degree of inconsistency. Therefore, by randomly removing one of the two vertices, we obtain a different MUS but all possible MUSs are minimal. The repair is then always maximal and is obtained by removing the resulting MUS from the ABox.

5.3 Best Temporal Repair in TKBs

Clearly, not just any repair is useful or interesting in the temporal setting. For instance, repairs that return only ancient assertions might be unwanted. However, in the light of temporal knowledge bases, we are interested in defining a strong notion of temporal repair. For such a purpose, we associate for each inconsistent assertion a temporal weight by taking the defining rigid axioms and the freshness of the assertion into consideration. Our aim is to guide the repair algorithm to remove assertions with the lowest temporal weight when they have the same degree of inconsistency.

▶ **Definition 13.** *We assign a temporal weight $w_t = t_i + \sigma$ for each inconsistent assertion a_i associated with the timestamp t_i. If a_i is an instance of a rigid predicate, σ expresses a duration for this rigid predicate. Otherwise, the temporal weight of a_i is based only on the timestamp t_i ($\sigma = 0$), so expresses the freshness of the assertion, as follow:*

$$w_t(a_i, t_i) = \begin{cases} t_i + \sigma & \text{if } a_i \in N_{RC} \cup N_G \\ t_i & \text{otherwise} \end{cases}$$

The intuition behind using a time range σ for each rigid predicate in repair phase is to set a maximum time threshold after which the assertion of this rigid predicate is discriminated or weakened. Figure 2 shows that the classic maximal repair in cases 2a and 2b could be the same because they share the same I_d. However, it is easy to see on a timeline that it is better to remove Minor(John, 2) in (2a) and Adult(John, 0) in (2b): the notion of temporal

weight is intended to capture situations where a maximal repair is temporally better than an other. At this level, we are considering a semi-automatic approach, which is guided by a user who will fix the value of σ by giving a duration to each defined rigid predicate according to his preferences, thus providing a repair that is as close as possible to the user needs.

6 Conclusion and Future Work

This paper provides a first exploration of repairing the ABox w.r.t a TBox defined over Temporal *DL-Lite*. The temporal language considered so far is the $T^{\mathbb{N}}$*DL-Lite* with which we can express and check several useful types of temporal constraints, such as defining temporal concepts in GCIs and rigid predicates, while maintaining good computational features. The first step in the reparation process is the detection of inconsistencies in *TDL-Lite* KBs. To do so, we proposed an equisatisfiable translation from *TDL-Lite* into *DL-Lite* KBs in order to use well optimized DL reasoners that include an axiom tracing service which allows extracting, from the TBox and the ABox, the relevant axioms involved in the inconsistency. This allows, in the second step, to perform a reparation based on the best repair semantic over *DL-Lite* ABoxes. We extended this semantic to the temporal setting by defining a temporal weight to guide the repair by removing, from the ABox, assertions with the highest inconsistency degree and the lowest temporal weight. Repair computation can be performed in polynomial time with respect to the number of inconsistent data assertions that appear in the ABox.

As a direction for our future work, we aim to enrich the definition of the TBox with General Concept Inclusions GCIs having temporal past operators on the right-hand side of the GCI. This could be equivalent to having future temporal operators on the left hand side of the GCI like $\{D \sqsubseteq C$, where D is a temporal concept$\}$. Let us note that in LTL some future temporal operators when expressed in the left hand side of an inclusion such as $(\Diamond_F\ q \rightarrow p)$ can be expressed as $(q \rightarrow \Box_P\ p)$ using a past operator on the right-hand side of the inclusion. More generally, we plan to investigate in practice repairing KBs based on multiple combinations of LTL with DL-Lite logics which are First Order rewritable [4]. Also, in the same spirit of the proposed temporal weight σ in section 5.3, which we defined as a temporal range for rigid predicates, we are considering adding metric operators to the *TDL-Lite* language [16, 7] that augment LTL temporal operators with time interval. Finally, it would be interesting to implement a repair framework and evaluate the scalability properties of our approach based on the temporal best repair semantics against temporal query answering under other semantics.

References

1 Andrea Acciarri, Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Mattia Palmieri, and Riccardo Rosati. Quonto: Querying ontologies. In *Proceedings, The 20th National Conference on Artificial Intelligence (AAAI 2005)*, pages 1670–1671, 2005.

2 Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proceedings of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems(PODS'99)*, pages 68–79, 1999.

3 Alessandro Artale and Enrico Franconi. Temporal description logics. In *Handbook of Temporal Reasoning in Artificial Intelligence*, pages 375–388. Elsevier, 2005.

4 Alessandro Artale, Roman Kontchakov, Alisa Kovtunova, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. Temporal OBDA with LTL and dl-lite. In *Informal Proceedings of the 27th International Workshop on Description Logics (DL 2014)*, CEUR W-S, pages 21–32, 2014.

5 Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. A cookbook for temporal conceptual data modelling with description logics. *ACM Trans. Comput. Log.*, 15(3):25:1–25:50, 2014.

6 F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

7 Franz Baader, Stefan Borgwardt, Patrick Koopmann, Ana Ozaki, and Veronika Thost. Metric temporal description logics with interval-rigid names. *ACM Trans. Comput. Log.*, 21(4):30:1–30:46, 2020.

8 Salima Benbernou and Mourad Ouziri. Enhancing data quality by cleaning inconsistent big RDF data. In *2017 IEEE International Conference on Big Data, BigData 2017, Boston, MA, USA, December 11-14, 2017*, pages 74–79, 2017.

9 Meghyn Bienvenu and Camille Bourgaux. Querying and repairing inconsistent prioritized knowledge bases: Complexity analysis and links with abstract argumentation. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*, pages 141–151, 2020.

10 Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI 2014)*, pages 996–1002, 2014.

11 Camille Bourgaux, Patrick Koopmann, and Anni-Yasmin Turhan. Ontology-mediated query answering over temporal and inconsistent data. *Semantic Web*, 10(3):475–521, 2019.

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

13 Jan Chomicki and Jerzy Marcinkowski. Minimal-change integrity maintenance using tuple deletions. *Inf. Comput.*, 197(1-2):90–121, 2005.

14 Stéphane Demri and Philippe Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Inf. Comput.*, 174(1):84–103, 2002.

15 Gianluigi Greco, Sergio Greco, and Ester Zumpano. A logical framework for querying and repairing inconsistent databases. *IEEE Trans. Knowl. Data Eng.*, 15(6):1389–1408, 2003.

16 Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Ana Ozaki. On metric temporal description logics. In *ECAI*, pages 837–845, 2016.

17 George Karakostas. A better approximation ratio for the vertex cover problem. *ACM Trans. Algorithms*, 5(4):41:1–41:8, 2009.

18 Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant semantics for description logics. In *Web Reasoning and Rule Systems - 4th International Conference*, volume 6333 of *Lecture Notes in Computer Science*, pages 103–117. Springer, 2010.

19 Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Query rewriting for inconsistent dl-lite ontologies. In *RR-2011 Proceedings*, volume 6902 of *Lecture Notes in Computer Science*, pages 155–169. Springer, 2011.

20 Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. Temporal description logics: A survey. In *TIME*, pages 3–14, 2008.

21 Giulia Masotti, Riccardo Rosati, and Marco Ruzzi. Practical abox cleaning in dl-lite (progress report). In *Proceedings of the 24th International Workshop on Description Logics(DL 2011)*, volume 745 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

22 A. Nakamura and H. Ono. On the size of refutation Kripke models for some linear modal and tense logics. *Studia Logica*, 39(4):325–333, 1980.

23 Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE Computer Society, 1977.

24 Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *J. Web Semant.*, 5(2):51–53, 2007.

25 Sabiha Tahrat, Germán Alejandro Braun, Alessandro Artale, Marco Gario, and Ana Ozaki. Automated reasoning in temporal dl-lite (extended abstract). In *Proceedings of the 33rd International Workshop on Description Logics (DL 2020)*, volume 2663 of *CEUR Workshop Proceedings*, 2020.