LTL over Description Logic Axioms

Franz Baader*, Silvio Ghilardi, and Carsten Lutz

[1] TU Dresden, Germany, baader@inf.tu-dresden.de
[2] Università degli Studi di Milano, Italy, ghilardi@dsi.unimi.it
[3] TU Dresden, Germany, lutz@inf.tu-dresden.de

1 Introduction

In many applications of Description Logics (DLs) [7], such as the use of DLs as ontology languages or conceptual modeling languages, being able to represent dynamic aspects of the application domain would be quite useful. This is, for instance, the case if one wants to use DLs as conceptual modeling languages for temporal databases [4]. Another example are medical ontologies, where the faithful representation of concepts would often require the description of temporal patterns. As a simple example, consider the concept "Concussion with no loss of consciousness," which is modeled as a primitive (i.e., not further defined) concept in the medical ontology SNOMED CT.[1] As argued in [18], a correct representation of this concept should actually say that, after the concussion, the patient remained conscious until the examination.

Since the expressiveness of pure DLs is not sufficient to describe such temporal patterns, a plethora of temporal extensions of DLs have been investigated in the literature.[2] These include approaches as diverse as the combination of DLs with Halpern and Shoham's logic of time intervals [17], formalisms inspired by action logics [1], the treatment of time points and intervals as a concrete domains [13], and the combination of standard DLs with standard (propositional) temporal logics into logics with a two-dimensional semantics, where one dimension is for time and the other for the DL domain [15, 19, 11]. In this paper, we follow the last approach, where we use the basic DL $\mathcal{ALC}$ [16] in the DL component and linear temporal logic (LTL) [14] (sometimes also called propositional temporal logic (PTL) [11]) in the temporal component. However, even after the DL and the temporal logic to be combined have been fixed, there remain several degrees of freedom when defining the resulting temporalized DL.

On the one hand, one must decide to which pieces of syntax temporal operators can be applied. Temporal operators may be allowed to be use as concept constructors, as required by the above example of a concussion with no loss of consciousness, which could be defined using the until-operator U of LTL as follows:

$$\exists \mathsf{finding}.\mathsf{Concussion} \sqcap \ \mathsf{Conscious}\ \mathsf{U}\ \exists \mathsf{procedure}.\mathsf{Examination}. \qquad (1)$$

* Supported by NICTA, Canberra Research Lab.

[1] see http://www.ihtsdo.org/our-standards/

[2] For a more thorough survey of the literature on temporalized DLs, see the technical report accompanying this paper [8] and the survey papers [2, 3, 12].

Alternatively or in addition, temporal operators may be applied to TBox axioms (i.e., general concept inclusions, GCIs) and/or to ABox assertions. For example, the temporalized TBox axiom

$$\Diamond\Box(\mathsf{UScitizen} \sqsubseteq \exists \mathsf{insured\_by}.\mathsf{HealthInsurer})$$

says that there is a future time point from which on US citizens will always have health insurance, and the formula Ψ:

$$\Diamond\big((\exists \mathsf{finding}.\mathsf{Concussion})(\mathsf{BOB}) \ \wedge \ \mathsf{Conscious}(\mathsf{BOB})\,\mathsf{U}\,(\exists \mathsf{procedure}.\mathsf{Examination})(\mathsf{BOB})\big) \tag{2}$$

says that, sometime in the future, Bob will have a concussion with no loss of consciousness between the concussion and the examination.

On the other hand, one must decide whether one wants to have rigid concepts and/or roles, i.e., concepts/roles whose interpretation does not vary over time. For example, the concept Human and the role has_father should probably be rigid since a human being will stay a human being and have the same father over his/her life-time, whereas Conscious should be a flexible concept (i.e., not rigid) since someone that is conscious at the moment need not always by conscious. Similarly, insured_by should be modeled as a flexible role. Using a logic that cannot enforce rigidity of concepts/roles may result in unintended models, and thus prevent certain useful inferences to be drawn. For example, the concept description $\exists \mathsf{has\_father}.\mathsf{Human} \sqcap \Diamond\,(\forall \mathsf{has\_father}.\neg\mathsf{Human})$ is only unsatisfiable if both has_father and Human are rigid.

The combination of (extensions of) $\mathcal{ALC}$ and LTL in which temporal operators can be applied to concept descriptions, TBox axioms, and ABox assertions has been studied by Wolter, Zakharyaschev, and others (see, e.g., [19, 11]). In particular, it is known that the basic reasoning problems are ExpSpace-complete. In this setting, rigid concepts can be defined, but rigid roles cannot. In fact, as also shown in [11], the addition of rigid roles causes undecidability even w.r.t. a global TBox (i.e., where the same TBox axioms must hold at all time points). Decidability can be regained by dropping TBoxes altogether, but the decision problem is still hard for non-elementary time. Decidable combinations of DLs and temporal logics that allow for rigid roles can be obtained by strongly restricting either the temporal component [6] or the DL component [5].

In this paper, we follow a different approach for regaining decidability in the presence of rigid roles: temporal operators are allowed to occur only in front of axioms (i.e., ABox assertions and TBox axioms), but not inside concept descriptions. We show that reasoning becomes simpler in this setting: with rigid roles, satisfiability is decidable (more precisely: 2-ExpTime-complete); without rigid roles, the complexity decreases to NExpTime-complete; and without any rigid symbols, it decreases further to ExpTime-complete (i.e., the same complexity as reasoning in $\mathcal{ALC}$ alone). We also consider another way of decreasing the complexity of satisfiability to ExpTime: satisfiability without rigid roles (but with rigid concepts) becomes ExpTime-complete if GCIs can occur only as global

axioms that must hold in every temporal world. Note that, in this case, ABox assertions are *not* assumed to be global, i.e., the valid ABox assertions may vary over time.

The situation we concentrate on in this paper (i.e., where temporal operators are allowed to occur only in front of axioms) has been considered before only for the case where there are no rigid concepts or roles. The combination approach introduced in [10] yields a decision procedure for this case, whose worst-case complexity is, however, non-optimal. Our ExpTime upper bound for this case actually also follows from more general results in [11] (see the remark following Theorem 14.15 on page 605 of [11]). However, also in [11], the setting where temporal operators are allowed to occur only in front of axioms is considered only in the absence of rigid symbols.

Obviously, the temporalized DLs we investigate in this paper cannot be used to define temporal concepts such as (1) for concussion with no loss of consciousness. However, they are nevertheless useful in ontology-based applications since they can be used to reason about a temporal sequence of ABoxes w.r.t. a (global or temporalized) TBox. For example, in an emergency ward, the vital parameters of a patient are monitored in short intervals (sometimes not longer than 10 minutes), and additional information is available from the patient record and added by doctors and nurses. Using concepts defined in a medical ontology like SNOMED CT, a high-level view of the medical status of the patient at a given time point can be given by an ABox. Obviously, the sequence of ABoxes obtained this way can be described using temporalized ABox assertions. Critical situations, which require the intervention of a doctor, can then be described by a formula in our temporalized DL, and recognized using the reasoning procedures developed in this paper. For example, given a formula ϕ encoding a sequence of ABoxes describing the medical status of Bob, starting at some time point t_0, and the formula ψ defined in (2), we can check whether Bob sometime after t_0 had a concussion with no loss of consciousness by testing $\phi \wedge \neg\psi$ for unsatisfiability.

2 Basic definitions

We assume that the reader is familiar with the basic DL $\mathcal{ALC}$ [16] and with the temporal logic LTL [14]. We consider general $\mathcal{ALC}$-TBoxes, i.e., TBoxes consist of finitely many *general concept inclusion axioms (GCIs)* of the form $C \sqsubseteq D$, where C, D are $\mathcal{ALC}$-concept descriptions. An ABox consists of a finite set of *assertions* of the form $a : C$ or $(a, b) : r$ where C is an $\mathcal{ALC}$-concept description, r is a role name, and a, b are individual names. We call both GCIs and ABox assertions *$\mathcal{ALC}$-axioms*. A Boolean combination of $\mathcal{ALC}$-axioms is called a *Boolean $\mathcal{ALC}$-knowledge base*. For LTL, we use the variant with a *non-strict until* (U) and a *next* (X) operator. We are now ready to define our new logic, called $\mathcal{ALC}$-LTL, where $\mathcal{ALC}$-axioms replace propositional letters in LTL.

Definition 1. *$\mathcal{ALC}$-LTL formulae are defined by induction:*

- *if α is an $\mathcal{ALC}$-axiom, then α is an $\mathcal{ALC}$-LTL formula;*

– *if* ϕ, ψ *are* $\mathcal{ALC}$*-LTL formulae, then so are* $\phi \wedge \psi$, $\phi \vee \psi$, $\neg\phi$, $\phi \mathsf{U} \psi$, *and* $\mathsf{X}\phi$.

As usual, we use true as an abbreviation for $A(a) \vee \neg A(a)$, $\Diamond\phi$ as an abbreviation for $\mathsf{true}\mathsf{U}\phi$ (*diamond*, which should be read as "sometime in the future"), and $\Box\phi$ as an abbreviation for $\neg\Diamond\neg\phi$ (*box*, which should be read as "always in the future"). The semantics of $\mathcal{ALC}$-LTL is based on $\mathcal{ALC}$-LTL structures, which are sequences of $\mathcal{ALC}$-interpretations over the same non-empty domain Δ (constant domain assumption). We assume that every individual name stands for a unique element of Δ (rigid individual names), and we make the unique name assumption (UNA), i.e., different individual names are interpreted by different elements of Δ.

Definition 2. *An* $\mathcal{ALC}$-LTL structure *is a sequence* $\mathfrak{I} = (\mathcal{I}_i)_{i=0,1,\ldots}$ *of* $\mathcal{ALC}$*-interpretations* $\mathcal{I}_i = (\Delta, \cdot^{\mathcal{I}_i})$ *obeying the UNA (called* worlds*) such that* $a^{\mathcal{I}_i} = a^{\mathcal{I}_j}$ *for all individual names* a *and all* $i, j \in \{0, 1, 2, \ldots\}$. *Given an* $\mathcal{ALC}$*-LTL formula* ϕ, *an* $\mathcal{ALC}$*-LTL structure* $\mathfrak{I} = (\mathcal{I}_i)_{i=0,1,\ldots}$, *and a time point* $i \in \{0, 1, 2, \ldots\}$, *validity of* ϕ *in* $\mathfrak{I}$ *at time* i *(written* $\mathfrak{I}, i \models \phi$*) is defined inductively:*

$$
\begin{array}{ll}
\mathfrak{I}, i \models C \sqsubseteq D & \text{iff } C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i} \\
\mathfrak{I}, i \models a : C & \text{iff } a^{\mathcal{I}_i} \in C^{\mathcal{I}_i} \\
\mathfrak{I}, i \models (a, b) : r & \text{iff } (a^{\mathcal{I}_i}, b^{\mathcal{I}_i}) \in r^{\mathcal{I}_i} \\
\mathfrak{I}, i \models \phi \wedge \psi & \text{iff } \mathfrak{I}, i \models \phi \text{ and } \mathfrak{I}, i \models \psi \\
\mathfrak{I}, i \models \phi \vee \psi & \text{iff } \mathfrak{I}, i \models \phi \text{ or } \mathfrak{I}, i \models \psi \\
\mathfrak{I}, i \models \neg\phi & \text{iff not } \mathfrak{I}, i \models \phi \\
\mathfrak{I}, i \models \mathsf{X}\phi & \text{iff } \mathfrak{I}, i+1 \models \phi \\
\mathfrak{I}, i \models \phi \mathsf{U} \psi & \text{iff there is } k \geq i \text{ such that } \mathfrak{I}, k \models \psi \\
 & \quad \text{and } \mathfrak{I}, j \models \phi \text{ for all } j, i \leq j < k
\end{array}
$$

As mentioned above, for some concepts and roles it is not desirable that their interpretation changes over time. Thus, we will sometimes assume that a subset of the set of concept and role names can be designated as being rigid. We will call the elements of this subset *rigid concept names* and *rigid role names*.

Definition 3. *We say that the* $\mathcal{ALC}$*-LTL structure* $\mathfrak{I} = (\mathcal{I}_i)_{i=0,1,\ldots}$ respects rigid concept names (role names) *iff* $A^{\mathcal{I}_i} = A^{\mathcal{I}_j}$ $(r^{\mathcal{I}_i} = r^{\mathcal{I}_j})$ *holds for all* $i, j \in \{0, 1, 2, \ldots\}$ *and all rigid concept names* A *(rigid role names* r*).*

3 The satisfiability problem in $\mathcal{ALC}$-LTL

Depending on whether rigid concept and role names are considered or not, we obtain different variants of the satisfiability problem.

Definition 4. *Let* ϕ *be an* $\mathcal{ALC}$*-LTL formula and assume that a subset of the set of concept and role names has been designated as being rigid.*

– *We say that* ϕ *is* satisfiable w.r.t. rigid names *iff there is an* $\mathcal{ALC}$*-LTL structure* $\mathfrak{I}$ *respecting rigid concept and role names such that* $\mathfrak{I}, 0 \models \phi$.

	W.r.t. rigid names	W.r.t. rigid concepts	Without rigid names
$\mathcal{ALC}$-LTL	2-ExpTime-complete	NExpTime-complete	ExpTime-complete
$\mathcal{ALC}$-LTL$\vert_{gGCI}$	2-ExpTime-complete	ExpTime-complete	ExpTime-complete

Table 1. Complexity of the satisfiability problem in $\mathcal{ALC}$-LTL and $\mathcal{ALC}$-LTL$\vert_{gGCI}$.

- *We say that ϕ is* satisfiable w.r.t. rigid concepts *iff there is an $\mathcal{ALC}$-LTL structure $\mathfrak{I}$ respecting rigid concept names such that $\mathfrak{I}, 0 \models \phi$.*
- *We say that ϕ is* satisfiable without rigid names *(or simply* satisfiable*) iff there is an $\mathcal{ALC}$-LTL structure $\mathfrak{I}$ such that $\mathfrak{I}, 0 \models \phi$.*

In this paper, we show that the complexity of the satisfiability problem for $\mathcal{ALC}$-LTL strongly depends on which of the above cases one considers. Note that it does not really make sense to consider satisfiability w.r.t. rigid role names, but without rigid concept names, as a separate case when investigating the complexity of the satisfiability problem. In fact, rigid concepts can be simulated by rigid roles: just introduce a new rigid role name r_A for each rigid concept name A, and then replace A by $\exists r_A.\top$.

Another dimension that influences the complexity of the satisfiability problem is whether GCIs occur globally or locally in the formula. Intuitively, a GCI occurs globally if it must hold in every world of the $\mathcal{ALC}$-LTL structure.

Definition 5. *We say that ϕ is an $\mathcal{ALC}$-LTL formula* with global GCIs *iff it is of the form $\phi = \Box\mathcal{B} \wedge \varphi$ where $\mathcal{B}$ is a conjunction of $\mathcal{ALC}$-axioms and φ is an $\mathcal{ALC}$-LTL formula that does not contain GCIs. We denote the fragment of $\mathcal{ALC}$-LTL that contains only $\mathcal{ALC}$-LTL formulae with global GCIs by $\mathcal{ALC}$-LTL$\vert_{gGCI}$.*

Note that saying, in the above definition, that $\mathcal{B}$ is a *conjunction* of $\mathcal{ALC}$-axioms just means that $\mathcal{B}$ is a TBox together with an ABox. We could have restricted $\mathcal{B}$ to being a conjunction of GCIs (i.e., a TBox) since assertions α in $\mathcal{B}$ could be moved as conjuncts $\Box\alpha$ to φ.[3] However, it turns out to be more convenient to allow also ABox assertions to occur in the "global part" $\Box\mathcal{B}$ of ϕ. Also note that it is important to restrict $\mathcal{B}$ to being a *conjunction* of $\mathcal{ALC}$-axioms rather than an arbitrary Boolean $\mathcal{ALC}$-knowledge base. In fact, the lower complexity for the case of satisfiability w.r.t. rigid concepts obtained in this case (see Table 1) would not hold without this restriction (see Corollary 6.8 in [8]).

Table 1 summarizes the results of our investigation of the complexity of the satisfiability problem in $\mathcal{ALC}$-LTL and its fragments.

4 Reasoning with rigid names

In this section, we investigate the complexity of the satisfiability problem in $\mathcal{ALC}$-LTL and $\mathcal{ALC}$-LTL$\vert_{gGCI}$ if rigid concepts and roles are available.

[3] This is the reason why we talk about $\mathcal{ALC}$-LTL formulae *with global GCIs* in this case, rather than about $\mathcal{ALC}$-LTL formulae with global axioms.

Theorem 1. *Satisfiability w.r.t. rigid names is* 2-ExpTime*-complete both in $\mathcal{ALC}$-LTL and in $\mathcal{ALC}$-LTL$|_{gGCI}$.*

2-ExpTime-hardness for satisfiability w.r.t. rigid names and with global GCIs (i.e., in $\mathcal{ALC}$-LTL$|_{gGCI}$) can be shown by a (quite intricate) reduction of the word problem for exponentially space bounded alternating Turing machines (see [8]). Obviously, this also yields 2-ExpTime-hardness for the more general case with arbitrary GCIs (i.e., in $\mathcal{ALC}$-LTL).

In the following, we prove the complexity upper bound for $\mathcal{ALC}$-LTL. Obviously, this also establishes the same upper bound for the restricted case of $\mathcal{ALC}$-LTL$|_{gGCI}$. Thus, let ϕ be an $\mathcal{ALC}$-LTL formula to be tested for satisfiability w.r.t. rigid names. We build its propositional abstraction $\widehat{\phi}$ by replacing each $\mathcal{ALC}$-axiom by a propositional variable such that there is a 1–1 relationship between the $\mathcal{ALC}$-axioms $\alpha_1, \ldots, \alpha_n$ occurring in ϕ and the propositional variables $p_1, \ldots, p_n$ used for the abstraction. We assume in the following that p_i was used to replace α_i $(i = 1, \ldots, n)$.

Consider a set $\mathcal{S} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$, i.e., a set of subsets of $\{p_1, \ldots, p_n\}$. Such a set induces the following (propositional) LTL formula:

$$\widehat{\phi}_{\mathcal{S}} := \widehat{\phi} \wedge \Box \left(\bigvee_{X \in \mathcal{S}} \left(\bigwedge_{p \in X} p \wedge \bigwedge_{p \notin X} \neg p \right) \right)$$

If ϕ is satisfiable in an $\mathcal{ALC}$-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i=0,1,\ldots}$, then there is an $\mathcal{S} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ such that $\widehat{\phi}_{\mathcal{S}}$ is satisfiable in a propositional LTL structure. In fact, for each $\mathcal{ALC}$-interpretation $\mathcal{I}_i$ of $\mathfrak{I}$, we define the set

$$X_i := \{p_j \mid 1 \leq j \leq n \text{ and } \mathcal{I}_i \text{ satisfies } \alpha_j\},$$

and then take $\mathcal{S} = \{X_i \mid i = 0, 1, \ldots\}$. The fact that $\mathfrak{I}$ satisfies ϕ implies that its propositional abstraction satisfies $\widehat{\phi}_{\mathcal{S}}$, where the propositional abstraction $\widehat{\mathfrak{I}} = (w_i)_{i=0,1,\ldots}$ of $\mathfrak{I}$ is defined such that world w_i makes variable p_j true iff $\mathcal{I}_i$ satisfies α_j. However, guessing such a set $\mathcal{S} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ and then testing whether the induced propositional LTL formula $\widehat{\phi}_{\mathcal{S}}$ is satisfiable is not sufficient for checking satisfiability w.r.t. rigid names of the $\mathcal{ALC}$-LTL formula ϕ. We must also check whether the guessed set $\mathcal{S}$ can indeed be induced by some $\mathcal{ALC}$-LTL structure that respects the rigid concept and role names.

To this purpose, assume that a set $\mathcal{S} = \{X_1, \ldots, X_k\} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ is given. For every $i, 1 \leq i \leq k$, and every *flexible* concept name A (*flexible* role name r) occurring in $\alpha_1, \ldots, \alpha_n$, we introduce a copy $A^{(i)}$ ($r^{(i)}$). We call $A^{(i)}$ ($r^{(i)}$) the ith copy of A (r). The $\mathcal{ALC}$-axiom $\alpha_j^{(i)}$ is obtained from α_j by replacing every occurrence of a flexible name by its ith copy. The sets X_i $(1 \leq i \leq k)$ induce the following Boolean $\mathcal{ALC}$-knowledge bases:

$$\mathcal{B}_i := \bigwedge_{p_j \in X_i} \alpha_j^{(i)} \wedge \bigwedge_{p_j \notin X_i} \neg \alpha_j^{(i)}$$

Lemma 1. *The $\mathcal{ALC}$-LTL formula ϕ is satisfiable w.r.t. rigid names iff there is a set $\mathcal{S} = \{X_1, \ldots, X_k\} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ such that the propositional LTL formula $\widehat{\phi}_{\mathcal{S}}$ is satisfiable and the Boolean $\mathcal{ALC}$-knowledge base $\mathcal{B} := \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ is consistent.*

A detailed proof of this lemma can be found in [8]. It remains to show that it provides us with a decision procedure for satisfiability in $\mathcal{ALC}$-LTL w.r.t. rigid names that runs in deterministic double-exponential time.

First, note that there are 2^{2^n} many subsets $\mathcal{S}$ of $\mathcal{P}(\{p_1, \ldots, p_n\})$ to be tested, where n is of course linearly bounded by the size of ϕ. For each of these subsets $\mathcal{S} = \{X_1, \ldots, X_k\}$, whose cardinality k is bounded by 2^n, we need to check satisfiability of $\widehat{\phi}_{\mathcal{S}}$ and consistency of $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$.

The size of $\widehat{\phi}_{\mathcal{S}}$ is at most exponential in the size of ϕ, and the complexity of the satisfiability problem in propositional LTL is in PSpace, and thus in particular in ExpTime. Consequently, satisfiability of $\widehat{\phi}_{\mathcal{S}}$ can be tested in double-exponential time in the size of ϕ.

The Boolean $\mathcal{ALC}$-knowledge base $\mathcal{B}$ is a conjunction of $k \leq 2^n$ Boolean $\mathcal{ALC}$-knowledge bases $\mathcal{B}_i$, where the size of each $\mathcal{B}_i$ is polynomial in the size of ϕ. The consistency problem for Boolean $\mathcal{ALC}$-knowledge base is ExpTime-complete (see, e.g., Theorem 2.27 in [11]). Consequently, consistency of $\mathcal{B}$ can also be tested in double-exponential time in the size of the input formula ϕ.

Overall, we thus have double-exponentially many tests, where each test takes double-exponential time. This provides us with a double-exponential bound for testing satisfiability in $\mathcal{ALC}$-LTL w.r.t. rigid names based on Lemma 1.

5 Reasoning with rigid concepts

In this section, we consider the case where rigid concept names are available, but not rigid role names. First, note that, in contrast to temporal DLs where temporal operator may occur inside of concept descriptions, rigid concept names cannot easily be expressed within the logic without rigid concept names. In fact, the GCIs $A \sqsubseteq \Box A$ and $\neg A \sqsubseteq \Box \neg A$ express that A must be interpreted in a rigid way. However, they are not allowed by the syntax of $\mathcal{ALC}$-LTL since the box is applied directly to a concept, and not to an axiom. We will show below that, for $\mathcal{ALC}$-LTL, the presence of rigid concept names indeed increases the complexity of the satisfiability problem, unless GCIs are restricted to being global. First, we treat the case of arbitrary GCIs, and then the special case of global GCIs.

Theorem 2. *Satisfiability in $\mathcal{ALC}$-LTL w.r.t. rigid concepts is* NExpTime*-complete.*

A detailed proof of the lower bound can be found in [8]. In the proof of the upper bound, we want to reuse Lemma 1. If we apply this lemma in the case where only concept names can be rigid, then we know that the Boolean $\mathcal{ALC}$-knowledge bases $\mathcal{B}_i$ are built over disjoint sets of role names. The only

shared names are the rigid concept names. Obviously, we can guess a set $\mathcal{S} = \{X_1, \ldots, X_k\} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$, within NExpTime. However, there are *two obstacles* on our way to a NExpTime decision procedure.

First, the propositional LTL formula $\widehat{\phi}_{\mathcal{S}}$ is of size exponential in the size of ϕ. Thus, a direct application of the PSpace decision procedure for satisfiability in propositional LTL would only yield an ExpSpace upper bound, which is not good enough. However, note that the only effect of the box-formula in $\widehat{\phi}_{\widehat{\mathcal{S}}}$ is to restrict the worlds w in a propositional LTL structure satisfying $\widehat{\phi}$ to being induced by one of the elements of $\widehat{\mathcal{S}}$. One way of deciding satisfiability of a propositional LTL formula $\widehat{\phi}$ is to construct a Büchi automaton $\mathcal{A}_{\widehat{\phi}}$ that accepts the propositional LTL structures satisfying $\widehat{\phi}$. To be more precise, let $\Sigma := \mathcal{P}(\{p_1, \ldots, p_n\})$. Then a given propositional LTL structure $\widehat{\mathfrak{I}} = (w_\iota)_{\iota=0,1,\ldots}$ can be represented by an infinite word $X_0 X_1 \ldots$ over Σ, where X_ι consists of the propositional variables that w_ι makes true. The Büchi automaton $\mathcal{A}_{\widehat{\phi}}$ is built such that it accepts exactly those infinite words over Σ that represent propositional LTL structures satisfying $\widehat{\phi}$. Consequently, $\widehat{\phi}$ is satisfiable iff the language accepted by $\mathcal{A}_{\widehat{\phi}}$ is non-empty. The size of $\mathcal{A}_{\widehat{\phi}}$ is exponential in the size of $\widehat{\phi}$, and the emptiness test for Büchi automata is polynomial in the size of the automaton. The automaton $\mathcal{A}_{\widehat{\phi}}$ can now easily be modified into one accepting exactly the words representing propositional LTL structures satisfying $\widehat{\phi}_{\widehat{\mathcal{S}}}$. In fact, we just need to remove all transitions that use a letter from $\Sigma \setminus \widehat{\mathcal{S}}$. Obviously, this modification can be done in time polynomial in the size of $\mathcal{A}_{\widehat{\phi}}$, and thus in time exponential in the size of $\widehat{\phi}$. The size of the resulting automaton is obviously still only exponential in the size of $\widehat{\phi}$, and thus its emptiness can be tested in time exponential in the size of $\widehat{\phi}$ (and hence of ϕ).

The second obstacle is the fact that $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ is of exponential size, and thus testing it directly for consistency using the known ExpTime decision procedure for satisfiability of Boolean $\mathcal{ALC}$-knowledge bases would provide us with a double-exponential time bound. Instead of testing the consistency of $\mathcal{B}$ directly we reduce this test to k separate consistency tests, each requiring time exponential in the size of ϕ. Before we can do this, we need another guessing step. Assume that $A_1, \ldots, A_r$ are all the rigid concept names occurring in ϕ, and that $a_1, \ldots, a_s$ are all the individual names occurring in ϕ. We guess a set $\mathcal{T} \subseteq \mathcal{P}(\{A_1, \ldots, A_r\})$ and a mapping $\mathfrak{t} : \{a_1, \ldots, a_s\} \to \mathcal{T}$. Again, this guess can clearly be done within NExpTime.

Given $\mathcal{T}$ and $\mathfrak{t}$, we extend the knowledge bases $\mathcal{B}_i$ to knowledge bases $\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t})$ as follows. For $Y \subseteq \{A_1, \ldots, A_r\}$, let C_Y be the concept description

$$C_Y := \bigsqcap_{A \in Y} A \sqcap \bigsqcap_{A \notin Y} \neg A.$$

We define $\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t})$ as

$$\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t}) := \mathcal{B}_i \wedge \bigwedge_{\mathfrak{t}(a) = Y} a : C_Y \wedge \top \sqsubseteq \bigsqcup_{Y \in \mathcal{T}} C_Y \wedge \bigwedge_{Y \in \mathcal{T}} \neg(\top \sqsubseteq \neg C_Y).$$

Lemma 2. *The Boolean $\mathcal{ALC}$-knowledge base $\mathcal{B} := \bigwedge_{1\leq i\leq k} \mathcal{B}_i$ is consistent iff there is a set $\mathcal{T} \subseteq \mathcal{P}(\{A_1, \ldots, A_r\})$ and a mapping $\mathfrak{t} : \{a_1, \ldots, a_s\} \to \mathcal{T}$ such that the Boolean knowledge bases $\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t})$ for $i = 1, \ldots, k$ are separately consistent.*

A proof of this lemma can be found in [8]. To finish the proof of Theorem 2, we must show that the consistency of $\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t})$ can be decided in time exponential in the size of the input formula ϕ. Note that this is not trivial. In fact, while the size of $\mathcal{B}_i \quad \wedge \quad \bigwedge_{\mathfrak{t}(a)=Y} a : C_Y$ is polynomial in the size of ϕ, the cardinality of $\mathcal{T}$, and thus the size of

$$\top \sqsubseteq \bigsqcup_{Y\in\mathcal{T}} C_Y \quad \wedge \quad \bigwedge_{Y\in\mathcal{T}} \neg(\top \sqsubseteq \neg C_Y), \tag{3}$$

can be exponential in the size of ϕ. Decidability of the consistency of $\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t})$ in time exponential in the size of ϕ is, however, an immediate consequence of the next lemma. To formulate this lemma, we need to introduce one more notation. Let $\widehat{\mathcal{B}}$ be a Boolean $\mathcal{ALC}$-knowledge base of size n, $A_1, \ldots, A_r$ concept names occurring in $\widehat{\mathcal{B}}$, and $\mathcal{T} \subseteq \mathcal{P}(\{A_1, \ldots, A_r\})$. Note that this implies that the cardinality of $\mathcal{T}$ is at most exponential in n, and the size of each $Y \in \mathcal{T}$ is linear in n. We say that $\widehat{\mathcal{B}}$ *is consistent w.r.t.* $\mathcal{T}$ iff it has a model that is also a model of (3). The following lemma can be shown by an adaptation of the proof of Theorem 2.27 in [11], which shows that the consistency problem for Boolean $\mathcal{ALC}$-knowledge bases is in ExpTime (see [8] for details).

Lemma 3. *Let $\widehat{\mathcal{B}}$ be a Boolean $\mathcal{ALC}$-knowledge base of size n, $A_1, \ldots, A_r$ concept names occurring in $\widehat{\mathcal{B}}$, and $\mathcal{T} \subseteq \mathcal{P}(\{A_1, \ldots, A_r\})$. Then, consistency of $\widehat{\mathcal{B}}$ w.r.t. $\mathcal{T}$ can be decided in time exponential in n.*

Overall, this completes the proof of Theorem 2. In fact, after two NExpTime guesses, all we have to do are k (i.e., exponentially many) ExpTime consistency tests.

Restricting GCIs to global ones decreases the complexity of the satisfiability problem.

Theorem 3. *Satisfiability in $\mathcal{ALC}$-LTL$|_{gGCI}$ w.r.t. rigid concepts is* ExpTime*-complete.*

ExpTime-hardness is an easy consequence of the well-known fact that concept satisfiability in $\mathcal{ALC}$ w.r.t. a single GCI is ExpTime-complete: C is satisfiable w.r.t. $D_1 \sqsubseteq D_2$ iff $a : C \wedge \Box(D_1 \sqsubseteq D_2)$ is satisfiable.

To prove the ExpTime upper bound, we consider an $\mathcal{ALC}$-LTL formula $\phi = \Box\mathcal{B} \wedge \varphi$, where $\mathcal{B}$ is a conjunction of $\mathcal{ALC}$-axioms and φ is an $\mathcal{ALC}$-LTL formula that does not contain GCIs. We say that $\mathcal{B}$ is *ϕ-exhaustive* if, for every individual name a and every rigid concept name A, either $a : A$ or $a : \neg A$ occurs as a conjunct in $\mathcal{B}$. We can assume without loss of generality that $\mathcal{B}$ is ϕ-exhaustive. In fact, given an arbitrary Boolean $\mathcal{ALC}$-knowledge base $\mathcal{B}$, we can build all the

ϕ-exhaustive knowledge bases $\mathcal{B}'$ that are obtained from $\mathcal{B}$ by conjoining to it, for every individual name a and every rigid concept name A, either $a : A$ or $a : \neg A$. Obviously, $\phi = \Box\mathcal{B} \wedge \varphi$ is satisfiable w.r.t. rigid concepts iff $\Box\mathcal{B}' \wedge \varphi$ is satisfiable w.r.t. rigid concepts for one of the extension $\mathcal{B}'$ of $\mathcal{B}$ obtained this way. Since the size of each such an extension is polynomial and there are only exponentially many such extensions, it is sufficient to show that testing satisfiability of $\Box\mathcal{B}' \wedge \varphi$ w.r.t. rigid concepts for ϕ-exhaustive knowledge bases $\mathcal{B}'$ is in ExpTime.

Following the approach used in the proof of Theorem 1, we abstract every ABox assertion α_i occurring in φ by a propositional variable p_i, thus building the propositional LTL-formula $\widehat{\varphi}$. Next, we compute the set $\widehat{\mathcal{S}}$, which consists of those $X \subseteq \{p_1, \ldots, p_n\}$ for which the Boolean $\mathcal{ALC}$-knowledge base

$$\mathcal{B}_X \quad := \quad \mathcal{B} \wedge \bigwedge_{p_j \in X} \alpha_j \wedge \bigwedge_{p_j \notin X} \neg\alpha_j$$

is consistent. This computation can be done in exponential time since it requires exponentially many ExpTime consistency tests.

Lemma 4. *Let $\phi = \Box\mathcal{B} \wedge \varphi$ be such that $\mathcal{B}$ is a ϕ-exhaustive conjunction of $\mathcal{ALC}$-axioms and φ is an $\mathcal{ALC}$-LTL formula not containing GCIs. Then ϕ is satisfiable w.r.t. rigid concepts iff the propositional LTL formula*

$$\widehat{\varphi}_{\widehat{\mathcal{S}}} \quad := \quad \widehat{\varphi} \wedge \Box\Big(\bigvee_{X \in \widehat{\mathcal{S}}} \Big(\bigwedge_{p_j \in X} p_j \wedge \bigwedge_{p_j \notin X} \neg p_j\Big)\Big)$$

is satisfiable.

The proof of this lemma can again be found in [8]. Note that this actually completes the proof of Theorem 3. In fact, as shown in the proof of Theorem 2, satisfiability of $\widehat{\varphi}_{\widehat{\mathcal{S}}}$ can be decided in exponential time.

6 Reasoning without rigid names

In this section, we consider the case where we have no rigid names at all.

Theorem 4. *Satisfiability without rigid names in $\mathcal{ALC}$-LTL and in its fragment $\mathcal{ALC}$-LTL$|_{gGCI}$ is* ExpTime*-complete.*

ExpTime-hardness is again an easy consequence of the fact that concept satisfiability in $\mathcal{ALC}$ w.r.t. a single GCI is ExpTime-complete. As already mentioned in the introduction, the ExpTime upper bound follows from more general results proved in Chapter 11 of [11] (see the remark following Theorem 14.14 on page 605 of [11]). A direct proof of the upper bound, which is similar to the proof of Theorem 3, is given in [8].

References

1. Artale, A., and Franconi, E. 1998. A temporal description logic for reasoning about actions and plans. *JAIR* 9.
2. Artale, A., and Franconi, E. 2000. A survey of temporal extensions of description logics. *AMAI* 30.
3. Artale, A., and Franconi, E. 2001. Temporal description logics. In *Handbook of Time and Temporal Reasoning in AI*. The MIT Press.
4. Artale, A.; Franconi, E.; Wolter, F.; and Zakharyaschev, M. 2002. A temporal description logic for reasoning over conceptual schemas and queries. In *Proc. JELIA'2002*, Springer LNCS 2424.
5. Artale, A.; Kontchakov, R.; Lutz, C.; Wolter, F.; and Zakharyaschev, M. 2007. Temporalising tractable description logics. In *Proc. TIME'07*, IEEE Press.
6. Artale, A.; Lutz, C.; and Toman, D. 2007. A description logic of change. In *Proc. IJCAI'07*, AAAI Press.
7. Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.
8. Baader, F.; Ghilardi, S.; and Lutz, C. 2008. LTL over description logic axioms. LTCS-Report 08-01, TU Dresden, Germany. See http://lat.inf.tu-dresden.de/research/reports.html.
9. Blackburn, P.; de Rijke, M.; and Venema, Y. 2001. *Modal Logic*. Cambridge University Press.
10. Finger, M., and Gabbay, D. 1992. Adding a temporal dimension to a logic system. *JoLLI* 2.
11. Gabbay, D.; Kurusz, A.; Wolter, F.; and Zakharyaschev, M. 2003. *Many-dimensional Modal Logics: Theory and Applications*. Elsevier.
12. Lutz, C.; Wolter, F.; and Zakharyaschev, M. 2008. Temporal description logics: A survey. In *Proc. TIME'08*, IEEE Press.
13. Lutz, C. 2001. Interval-based temporal reasoning with general TBoxes. In *Proc. IJCAI'01*, AAAI Press.
14. Pnueli, A. 1977. The temporal logic of programs. In *Proc. FOCS'77*.
15. Schild, K. 1993. Combining terminological logics with tense logic. In *Proc. EPIA'93*, Springer LNCS 727.
16. Schmidt-Schauß, M., and Smolka, G. 1991. Attributive concept descriptions with complements. *AIJ* 48.
17. Schmiedel, A. 1990. A temporal terminological logic. In *Proc. AAAI'90*, AAAI Press.
18. Schulz, S.; Markó, K.; and Suntisrivaraporn, B. 2006. Complex occurrents in clinical terminologies and their representation in a formal language. In *Proc. 1st European Conference on SNOMED CT (SMCS'06)*.
19. Wolter, F., and Zakharyaschev, M. 1999. Temporalizing description logics. In *Proc. FroCoS'98*, Wiley.