LTL over Description Logic Axioms

Franz Baader*
TU Dresden, Germany
baader@inf.tu-dresden.de

Silvio Ghilardi
Università degli Studi di Milano, Italy
ghilardi@dsi.unimi.it

Carsten Lutz
TU Dresden, Germany
lutz@inf.tu-dresden.de

Abstract

Most of the research on temporalized Description Logics (DLs) has concentrated on the case where temporal operators can occur within DL concept descriptions. In this setting, reasoning usually becomes quite hard if rigid roles, i.e., roles whose interpretation does not change over time, are available. In this paper, we consider the case where temporal operators are allowed to occur only in front of DL axioms (i.e., ABox assertions and general concept inclusion axioms), but not inside of concepts descriptions. As the temporal component, we use linear temporal logic (LTL) and in the DL component we consider the basic DL $\mathcal{ALC}$. We show that reasoning in the presence of rigid roles becomes considerably simpler in this setting.

Introduction

In many applications of Description Logics (DLs) (Baader *et al.* 2003), 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 (Artale *et al.* 2002). 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 (Schulz, Markó, & Suntisrivaraporn 2006), 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 (Schmiedel 1990), formalisms inspired by action logics (Artale & Franconi 1998), the treatment of time points and intervals as a concrete domains (Lutz 2001), 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 (Schild 1993; Wolter & Zakharyaschev 1999; Gabbay *et al.* 2003). In this paper, we follow the last approach, where we use the basic DL $\mathcal{ALC}$ (Schmidt-Schauß & Smolka 1991) in the DL component and linear temporal logic (LTL) (Pnueli 1977) 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 which pieces of syntax temporal operators can be applied to. Temporal operators may be allowed to be used 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:

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

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

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

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

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

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 $\mathsf{has\_father}$ should

*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 (Baader, Ghilardi, & Lutz 2008) and the survey papers (Artale & Franconi 2000; 2001; Lutz, Wolter, & Zakharyaschev 2008).

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., (Wolter & Zakharyaschev 1999; Gabbay *et al.* 2003)). In particular, it is known that the basic reasoning problems such as satisfiability are EXPSPACE-complete. In this setting, rigid concepts can be defined, but rigid roles cannot. In fact, as shown in (Gabbay *et al.* 2003), 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 (Artale, Lutz, & Toman 2007) or the DL component (Artale *et al.* 2007).

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 two other ways of decreasing the complexity of satisfiability to EXPTIME. On the one hand, 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. On the other hand, satisfiability with rigid concepts and roles becomes EXPTIME-complete if the temporal component is restricted appropriately by replacing the temporal operators until (U) and next (X) of LTL with diamond ($\Diamond$), which expresses "sometime in the future."

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 (Finger & Gabbay 1992) 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 (Gabbay *et al.* 2003) (see the remark following Theorem 14.15 on page 605 there). However, also in (Gabbay *et al.* 2003), 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.

Basic definitions

The temporalized DL $\mathcal{ALC}$-LTL introduced in this paper combines the basic DL $\mathcal{ALC}$ with linear temporal logic (LTL). We start by recalling the relevant definitions for $\mathcal{ALC}$.

Definition 1. *Let N_C, N_R, and N_I respectively be disjoint sets of* concept names, role names, *and* individual names. *The set of $\mathcal{ALC}$-*concept descriptions *is the smallest set such that*

- *all concept names are $\mathcal{ALC}$-concept descriptions;*
- *if C, D are $\mathcal{ALC}$-concept descriptions and $r \in N_R$, then $\neg C$, $C \sqcup D$, $C \sqcap D$, $\exists r.C$, and $\forall r.C$ are $\mathcal{ALC}$-concept descriptions.*

A general concept inclusion axiom (GCI) *is of the form $C \sqsubseteq D$, where C, D are $\mathcal{ALC}$-concept descriptions, and an* assertion *is 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 assertions $\mathcal{ALC}$-*axioms. *A Boolean combination of $\mathcal{ALC}$-axioms is called a* Boolean $\mathcal{ALC}$-knowledge base, *i.e.,*

- *every $\mathcal{ALC}$-axiom is a Boolean $\mathcal{ALC}$-knowledge base;*
- *if $\mathcal{B}_1$ and $\mathcal{B}_2$ are Boolean $\mathcal{ALC}$-knowledge bases, then so are $\mathcal{B}_1 \wedge \mathcal{B}_2$, $\mathcal{B}_1 \vee \mathcal{B}_2$, and $\neg\mathcal{B}_1$.*

*An $\mathcal{ALC}$-*TBox *is a conjunction of GCIs, and an $\mathcal{ALC}$-*ABox *is a conjunction of assertions.*

According to this definition, TBoxes and ABoxes are special kinds of Boolean knowledge bases. However, note that

they are often written as sets of axioms rather than as conjunctions of these axioms. The semantics of $\mathcal{ALC}$ is defined through the notion of an interpretation.

Definition 2. *An* interpretation *is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where the* domain *$\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function that assigns to every concept name A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, to every role name r a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to every individual name a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. This function is extended to $\mathcal{ALC}$-concept descriptions as follows:*

- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}, (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}, (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$;
- $(\exists r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid$ *there is a* $y \in \Delta^{\mathcal{I}}$ *with* $(x, y) \in r^{\mathcal{I}}$ *and* $y \in C^{\mathcal{I}}\}$;
- $(\forall r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid$ *for all* $y \in \Delta^{\mathcal{I}}$, $(x, y) \in r^{\mathcal{I}}$ *implies* $y \in C^{\mathcal{I}}\}$.

The interpretation $\mathcal{I}$ is a model *of the $\mathcal{ALC}$-axioms $C \sqsubseteq D$, $a : C$, and $(a, b) : r$ iff it respectively satisfies $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$. The notion of a model is extended to Boolean $\mathcal{ALC}$-knowledge bases as follows:*

- *$\mathcal{I}$ is a model of $\mathcal{B}_1 \wedge \mathcal{B}_2$ iff it is a model of $\mathcal{B}_1$ and $\mathcal{B}_2$;*
- *$\mathcal{I}$ is a model of $\mathcal{B}_1 \vee \mathcal{B}_2$ iff it is a model of $\mathcal{B}_1$ or $\mathcal{B}_2$;*
- *$\mathcal{I}$ is a model of $\neg\mathcal{B}_1$ iff it is not a model of $\mathcal{B}_1$.*

We say that the Boolean $\mathcal{ALC}$-knowledge base $\mathcal{B}$ is consistent *iff it has a model. The concept description C is* satisfiable *w.r.t. the GCI $D_1 \sqsubseteq D_2$ iff there is a model $\mathcal{I}$ of $D_1 \sqsubseteq D_2$ with $C^{\mathcal{I}} \neq \emptyset$.*

In Description Logics, it is often assumed that the interpretations satisfy the *unique name assumption (UNA)*, i.e., different individual names are interpreted by different elements of the domain.

For LTL, we use the variant with a *non-strict until* (U) and a *next* (X) operator. Instead of first introducing the propositional temporal logic LTL, we directly define our new temporalized DL, called $\mathcal{ALC}$-LTL. The difference to LTL is that $\mathcal{ALC}$-axioms replace propositional letters.

Definition 3. *$\mathcal{ALC}$-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.

Definition 4. *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:*

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

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*. Concept and role names that do not belong to this subset will be called *flexible*.

Definition 5. *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).*

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 6. *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 $\mathfrak{I}, 0 \models \phi$.*
- *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 7. *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$|_{gGCI}$.

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 below) would not hold without this restriction (see Corollary 6.8 in (Baader, Ghilardi, & Lutz 2008)).

Instead of restricting to $\mathcal{ALC}$-LTL formulae with global GCIs, we can also restrict the temporal component, by considering the fragment $\mathcal{ALC}$-LTL$|_{\Diamond}$ of $\mathcal{ALC}$-LTL in which $\Diamond$ is the only temporal operator. In this fragment, neither U nor X is definable.

Definition 8. *$\mathcal{ALC}$-LTL$|_{\Diamond}$ formulae are defined by induction:*

- *if α is an $\mathcal{ALC}$-axiom, then α is an $\mathcal{ALC}$-LTL$|_{\Diamond}$ formula;*
- *if ϕ, ψ are $\mathcal{ALC}$-LTL$|_{\Diamond}$ formulae, then so are $\phi \wedge \psi$, $\phi \vee \psi$, $\neg\phi$, and $\Diamond\phi$.*

The semantics of $\mathcal{ALC}$-LTL$|_{\Diamond}$ formulae is defined as in the case of $\mathcal{ALC}$-LTL. In particular, the interpretation of the diamond operator is defined as

$$\mathfrak{I}, i \models \Diamond\phi \quad \text{iff} \quad \text{there is } k \geq i \text{ such that } \mathfrak{I}, k \models \phi.$$

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

Reasoning with rigid names

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

Theorem 9. *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 the Appendix). 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 $\mathcal{S} = \{X_i \mid i = 0, 1, \ldots\}$. We say that $\mathcal{S}$ is *induced* by the $\mathcal{ALC}$-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i=0,1,\ldots}$. 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 10. *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.*

Proof. For the "only if" direction, recall that we have already seen how an $\mathcal{ALC}$-LTL structure $\mathfrak{I} = (\mathcal{I}_\iota)_{\iota=0,1,\ldots}$ satisfying ϕ can be used to define a set $\mathcal{S} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ such that $\widehat{\phi}_{\mathcal{S}}$ is satisfiable. Let $\mathcal{S} = \{X_1, \ldots, X_k\}$. For each $\iota = 0, 1, \ldots$ there is an index $i_\iota \in \{1, \ldots, k\}$ such that $\mathcal{I}_\iota$ induces the set X_{i_ι}, i.e.,

$$X_{i_\iota} = \{p_j \mid 1 \leq j \leq n \text{ and } \mathcal{I}_\iota \text{ satisfies } \alpha_j\},$$

and, conversely, for each $i \in \{1, \ldots, k\}$ there is an index $\iota \in \{0, 1, 2, \ldots\}$ such that $i = i_\iota$. Let $\iota_1, \ldots, \iota_k \in \{0, 1, 2, \ldots\}$

[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.

	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$_{\mid gGCI}$	2-ExpTime-complete	ExpTime-complete	ExpTime-complete
$\mathcal{ALC}$-LTL$_{\mid \Diamond}$	ExpTime-complete	ExpTime-complete	ExpTime-complete

Table 1: Complexity of the satisfiability problem in $\mathcal{ALC}$-LTL and its fragments.

be such that $i_{\iota_1} = 1, \ldots, i_{\iota_k} = k$. The $\mathcal{ALC}$-interpretation $\mathcal{J}_i$ is obtained from $\mathcal{I}_{\iota_i}$ by interpreting the ith copy of each flexible name like the original flexible name, and by forgetting about the interpretations of the flexible names. By our construction of $\mathcal{J}_i$ and our definition of the Boolean $\mathcal{ALC}$-knowledge base $\mathcal{B}_i$, we have that $\mathcal{J}_i$ is a model of $\mathcal{B}_i$. Recall that the interpretations $\mathcal{I}_{\iota_1}, \ldots, \mathcal{I}_{\iota_k}$ (and thus also $\mathcal{J}_1, \ldots, \mathcal{J}_k$) all have the same domain. In addition, the interpretations of the rigid names coincide in $\mathcal{I}_{\iota_1}, \ldots, \mathcal{I}_{\iota_k}$ (and thus also in $\mathcal{J}_1, \ldots, \mathcal{J}_k$) and the flexible symbols have been renamed. Thus, the union $\mathcal{J}$ of $\mathcal{J}_1, \ldots, \mathcal{J}_k$ is a well-defined $\mathcal{ALC}$-interpretation, and it is easy to see that it is a model of $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$.

To show the "if" direction, assume that there is a set $\mathcal{S} = \{X_1, \ldots, X_k\} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ such that $\widehat{\phi}_{\mathcal{S}}$ is satisfiable and $\mathcal{B} := \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ is consistent. Let $\widehat{\mathfrak{I}} = (w_\iota)_{\iota=0,1,\ldots}$ be a propositional LTL structure satisfying $\widehat{\phi}_{\mathcal{S}}$, and let $\mathcal{J}$ be an $\mathcal{ALC}$-interpretation satisfying $\mathcal{B}$. By the definition of $\widehat{\phi}_{\mathcal{S}}$, for every world w_ι there is exactly one index $i_\iota \in \{1, \ldots, k\}$ such that w_ι satisfies

$$\bigwedge_{p \in X_{i_\iota}} p \wedge \bigwedge_{p \notin X_{i_\iota}} \neg p.$$

For $i \in \{1, \ldots, k\}$, we use the $\mathcal{ALC}$-interpretation $\mathcal{J}$ satisfying $\mathcal{B}$ to define an $\mathcal{ALC}$-interpretation $\mathcal{J}_i$ as follows: $\mathcal{J}_i$ interprets the rigid names like $\mathcal{J}$, and it interprets the flexible names just as $\mathcal{J}$ interprets the ith copies of them. Note that the interpretations $\mathcal{J}_i$ are over the same domain and respect the rigid symbols, i.e., they interpret them identically. We can now define an $\mathcal{ALC}$-LTL structure respecting rigid symbols and satisfying ϕ as follows: $\mathfrak{I} := (\mathcal{I}_\iota)_{\iota=0,1,\ldots}$ where $\mathcal{I}_\iota := \mathcal{J}_{i_\iota}$. ❏

It remains to show that this lemma 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 (Gabbay *et al.* 2003)). 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 10.

The hardness proof given in the Appendix shows that this double-exponential upper bound is indeed optimal. However, to get an intuition for what actually makes the problem so hard, let us analyze in more detail where the algorithm sketched above needs to spend double-exponential time. The algorithm needs

1. to consider 2^{2^n} many subsets $\mathcal{S} = \{X_1, \ldots, X_k\}$ of $\mathcal{P}(\{p_1, \ldots, p_n\})$;
2. for each such subset test $\widehat{\phi}_{\mathcal{S}}$ for satisfiability;
3. test $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ for consistency.

The main culprit is 3. Intuitively, the presence of rigid roles enforces that the consistency test for the conjunction $\bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ needs to be done for the whole conjunction, and cannot be reduced to separate test for the conjuncts (or some enriched versions of the conjuncts). Thus, the Boolean $\mathcal{ALC}$-knowledge base to be tested for consistency is exponentially large. Since the consistency problem for Boolean $\mathcal{ALC}$-knowledge bases is ExpTime-complete, testing this knowledge base for consistency requires in the worst-case double-exponential time. In contrast, 2. is actually harmless. According to what we have said above, it needs exponential space, but we will see in the next section that this can even be reduced to exponential time. Finally, 1. also does not really require double-exponential time since one could guess an appropriate set $\mathcal{S}$ within NExpTime.

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 operators 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 11. *Satisfiability in $\mathcal{ALC}$-LTL w.r.t. rigid concepts is* NEXPTIME*-complete.*

A detailed proof of the lower bound, by reduction of the 2^{n+1}-bounded domino problem (Börger, Grädel, & Gurevich 1997; Tobies 1999), can be found in (Baader, Ghilardi, & Lutz 2008). In the proof of the upper bound, we want to reuse Lemma 10. 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 $\widehat{\phi}_{\mathcal{S}}$. 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

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

Since we consider the case where only concept names can be rigid, we know that the Boolean $\mathcal{ALC}$-knowledge bases $\mathcal{B}_i$ are built over disjoint sets of role names. The only names shared among the knowledge bases $\mathcal{B}_i$ are the rigid concept names.

Lemma 12. *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.*

Proof. For the "only if" direction, assume that $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ has a model $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$. Let $\mathcal{T}$ consist of those sets $Y \subseteq \{A_1, \ldots, A_r\}$ such that there is a $d \in \Delta$ with $d \in (C_Y)^{\mathcal{I}}$, and let $\mathfrak{t}$ be the mapping satisfying $\mathfrak{t}(a) = Y$ iff $a^{\mathcal{I}} \in (C_Y)^{\mathcal{I}}$. It is easy to see that, with this choice of $\mathcal{T}$ and $\mathfrak{t}$, all the knowledge bases $\widehat{\mathcal{B}}_i(\mathcal{T}, \mathfrak{t})$ for $i = 1, \ldots, k$ have $\mathcal{I}$ as model.

To show the "if" direction, assume that 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 $i = 1, \ldots, k$ have models $\mathcal{I}_i = (\Delta_i, \cdot^{\mathcal{I}_i})$. We can assume without loss of generality[4] that

- the domains Δ_i are countably infinite, and
- in each model $\mathcal{I}_i$, the sets $Y \in \mathcal{T}$ are realized by countably infinitely many individuals, i.e., there are countably infinitely many elements $d \in \Delta_i$ such that $d \in (C_Y)^{\mathcal{I}_i}$.

Consequently, the domains Δ_i are partitioned into the countably infinite sets $\Delta_i(Y)$ (for $Y \in \mathcal{T}$), which are defined as follows:

$$\Delta_i(Y) := \{d \in \Delta_i \mid d \in (C_Y)^{\mathcal{I}_i}\}$$

In addition, for each individual name $a \in \{a_1, \ldots, a_s\}$ we have

$$a^{\mathcal{I}_i} \in \Delta_i(\mathfrak{t}(a))$$

We are now ready to define the model $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ of $\mathcal{B}$. As the domain of $\mathcal{I}$ we take the domain of $\mathcal{I}_1$, i.e., $\Delta := \Delta_1$. Accordingly, we define $\Delta(Y) := \Delta_1(Y)$ for all $Y \in \mathcal{T}$. Because of the properties stated above, there exist bijections $\pi_i : \Delta_i \to \Delta$ such that

[4] This is an easy consequence of the fact that Boolean $\mathcal{ALC}$-knowledge bases always have a finite model and that the countably infinite disjoint union of a model of a Boolean $\mathcal{ALC}$-knowledge base is again a model of this knowledge base.

- the restriction of π_i to $\Delta_i(Y)$ is a bijection between $\Delta_i(Y)$ and $\Delta(Y)$;
- π_i respects individual names, i.e., $\pi_i(a^{\mathcal{I}_i}) = a^{\mathcal{I}_1}$ holds for all $a \in \{a_1, \ldots, a_s\}$. (Note that we have the unique name assumption for individual names.)

We us these bijections to define the interpretation function $\cdot^{\mathcal{I}}$ of $\mathcal{I}$ as follows:

- If A is a flexible concept name, then $\mathcal{B}$ contains its copies $A^{(i)}$ for $i = 1, \ldots, k$. Their interpretation is defined as follows:
$$(A^{(i)})^{\mathcal{I}} := \{\pi(d) \mid d \in A^{\mathcal{I}_i}\}.$$
- All role names r are flexible, and $\mathcal{B}$ contains their copies $r^{(i)}$ for $i = 1, \ldots, k$. Their interpretation is defined as follows:
$$(r^{(i)})^{\mathcal{I}} := \{(\pi(d), \pi(e)) \mid (d, e) \in r^{\mathcal{I}_i}\}.$$
- If A is a rigid concept names, then we define
$$A^{\mathcal{I}} := A^{\mathcal{I}_1}$$
- If a is an individual name, then we define
$$a^{\mathcal{I}} := a^{\mathcal{I}_1}$$

To prove the lemma, it remains to show that $\mathcal{I}$ is a model of all the knowledge bases $\mathcal{B}_i$ ($i = 1, \ldots, k$). This is an immediate consequence of the fact that π_i is an isomorphism between $\mathcal{I}_i$ and $\mathcal{I}$ w.r.t. the concept and role names occurring in $\mathcal{B}_i$. The isomorphism condition is satisfied for flexible concepts and roles by our definition of $\cdot^{\mathcal{I}}$, and for individual names by our assumptions on π_i. Now, let A be a rigid concept name. We must show that $d \in A^{\mathcal{I}_i}$ iff $\pi_i(d) \in A^{\mathcal{I}}$ holds for all $d \in \Delta_i$. Since Δ_i is partitioned into the sets $\Delta_i(Y)$ for $Y \in \mathcal{T}$, we know that there is a $Y \in \mathcal{T}$ such that $d \in \Delta_i(Y)$, i.e., $d \in (C_Y)^{\mathcal{I}_i}$. In addition, we know that $\pi_i(d) \in \Delta(Y)$, i.e., $\pi_i(d) \in (C_Y)^{\mathcal{I}_1} = (C_Y)^{\mathcal{I}}$. This implies that $d \in A^{\mathcal{I}_i}$ iff $A \in Y$ iff $\pi_i(d) \in A^{\mathcal{I}}$, which finishes the proof that π_i is an isomorphism between $\mathcal{I}_i$ and $\mathcal{I}$. Consequently, $\mathcal{I}$ is a model of $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$, which in turn finishes the proof of the lemma. ❑

To complete the proof of Theorem 11, 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 $\mathcal{B}_i \ \wedge \ \bigwedge_{\mathfrak{t}(a)=Y} a : C_Y$ is polynomial in the size of ϕ, the cardinality of $\mathcal{T}$, and thus the size of

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

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 (Gabbay *et al.* 2003), which shows that the consistency problem for Boolean $\mathcal{ALC}$-knowledge bases is in EXPTIME (see (Baader, Ghilardi, & Lutz 2008) for details).

Lemma 13. *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 11. 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 14. *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 occurring in ϕ, 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 occurring in ϕ, 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 9, 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

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

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

Lemma 15. *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

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

is satisfiable.

The proof of this lemma can again be found in (Baader, Ghilardi, & Lutz 2008). Note that this actually completes the proof of Theorem 14. In fact, as shown in the proof of Theorem 11, satisfiability of $\widehat{\varphi}_{\widehat{\mathcal{S}}}$ can be decided in exponential time.

Reasoning without rigid names

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

Theorem 16. *Satisfiability without rigid names in $\mathcal{ALC}$-LTL and in $\mathcal{ALC}$-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 (Gabbay *et al.* 2003) (see the remark following Theorem 14.14 on page 605 of (Gabbay *et al.* 2003)). A direct proof of the upper bound, which is similar to the proof of Theorem 14, is given in (Baader, Ghilardi, & Lutz 2008).

Restricting the temporal component

In this section, we consider the fragment $\mathcal{ALC}$-LTL$|_\Diamond$ of $\mathcal{ALC}$-LTL, in which $\Diamond$ is the only temporal operator. Our aim is to show that satisfiability in $\mathcal{ALC}$-LTL$|_\Diamond$ w.r.t. rigid names is in EXPTIME. The main reason for this is that we can restrict the attention to $\mathcal{ALC}$-LTL structures respecting rigid concept and role names that consist of at most polynomially many distinct $\mathcal{ALC}$-interpretations. The *weight* of the $\mathcal{ALC}$-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i=0,1,\ldots}$ is defined to be the cardinality of the set $\{\mathcal{I}_i \mid i = 0, 1, \ldots\}$.[5]

Lemma 17. *Let ϕ be an $\mathcal{ALC}$-LTL$|_\Diamond$ formula of size m. If ϕ is satisfiable w.r.t. rigid names, then there is an $\mathcal{ALC}$-LTL structure $\mathfrak{I}$ respecting rigid concept and role names of weight at most $|\phi| + 2$ such that $\mathfrak{I}, 0 \models \phi$.*

The proof of this fact, which can be found in (Baader, Ghilardi, & Lutz 2008), is a straightforward generalization to $\mathcal{ALC}$-LTL$|_\Diamond$ of a very similar proof for LTL$|_\Diamond$, the restriction of propositional LTL to its diamond fragment (see, e.g., Lemma 6.40 in (Blackburn, de Rijke, & Venema 2001)). Based on this lemma, the following modified version of Lemma 10 can be shown.

Lemma 18. *Let ϕ be an $\mathcal{ALC}$-LTL$|_\Diamond$ formula of size m. Then, ϕ is satisfiable w.r.t. rigid names iff there is a set $\mathcal{S} = \{X_1, \ldots, X_k\} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ of cardinality $k \leq m+2$ such that the propositional LTL$|_\Diamond$ 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.*

Proof. The "if" direction of this lemma is an immediate consequence of Lemma 10. For the "only if" direction, we can use the proof of the "only if" direction Lemma 10. The only difference is that we start with an $\mathcal{ALC}$-LTL structure $\mathfrak{I}$ respecting rigid concept and role names *of weight at most* $|\phi| + 2$ such that $\mathfrak{I}, 0 \models \phi$. The existence of such a structure is guaranteed by Lemma 17. It is easy to see that then the set $\mathcal{S} = \{X_1, \ldots, X_k\}$ induced by this structure is indeed of cardinality $k \leq |\phi| + 2$. ❑

It remains to show that this lemma provides us with a decision procedure for satisfiability in $\mathcal{ALC}$-LTL$|_\Diamond$ w.r.t. rigid names that runs in deterministic exponential time. There are $\leq 2^{m(m+2)}$ subsets $\mathcal{S} \subseteq \mathcal{P}(\{p_1, \ldots, p_n\})$ of cardinality $\leq m + 2$ to be considered, and the size of each such subset $\mathcal{S} = \{X_1, \ldots, X_k\}$ is polynomial in m. Thus, the size of both $\widehat{\phi}_{\mathcal{S}}$ and $\mathcal{B} = \bigwedge_{1 \leq i \leq k} \mathcal{B}_i$ is polynomial in m. Since satisfiability in propositional LTL is in PSPACE and the consistency problems for Boolean $\mathcal{ALC}$-knowledge bases is in EXPTIME, this shows that satisfiability in $\mathcal{ALC}$-LTL$|_\Diamond$ w.r.t. rigid names is in EXPTIME.

EXPTIME-hardness of satisfiability in $\mathcal{ALC}$-LTL$|_\Diamond$ (even without any rigid names) is again an easy consequence of the fact that concept satisfiability in $\mathcal{ALC}$ w.r.t. a single GCI is EXPTIME-complete.

Theorem 19. *Satisfiability in $\mathcal{ALC}$-LTL$|_\Diamond$ w.r.t. rigid names is an* EXPTIME*-complete problem. The same is true for satisfiability w.r.t. rigid concepts and without rigid names.*

Conclusion

The faithful modelling of dynamically changing environments with a temporalized DL often requires the availability of rigid concepts and roles. We have shown that decidability and an elementary complexity upper bound can be achieved also in the presence of rigid roles by restricting the application of temporal operators to DL axioms. This is a big advance over the case where temporal operators can occur inside concept descriptions, in which rigid roles cause undecidability in the presence of a TBox and hardness for non-elementary time even without a TBox.

The decision procedures we have described in this paper were developed for the purpose of showing worst-case complexity upper bounds. The major topic for future work is to optimize them such that they can be used in practice, where we will first concentrate on the application scenario sketched in the introduction.

References

Artale, A., and Franconi, E. 1998. A temporal description logic for reasoning about actions and plans. *JAIR* 9.

Artale, A., and Franconi, E. 2000. A survey of temporal extensions of description logics. *AMAI* 30.

[5] Recall that all the $\mathcal{ALC}$-interpretations within one $\mathcal{ALC}$-LTL structure have the same domain. For this reason, we can use exact equality of interpretations rather than equality up to isomorphism when defining the weight of an $\mathcal{ALC}$-LTL structure.

Artale, A., and Franconi, E. 2001. Temporal description logics. In *Handbook of Time and Temporal Reasoning in AI*. The MIT Press.

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.

Artale, A.; Kontchakov, R.; Lutz, C.; Wolter, F.; and Zakharyaschev, M. 2007. Temporalising tractable description logics. In *Proc. TIME'07*, IEEE Press.

Artale, A.; Lutz, C.; and Toman, D. 2007. A description logic of change. In *Proc. IJCAI'07*, AAAI Press.

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.

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.

Blackburn, P.; de Rijke, M.; and Venema, Y. 2001. *Modal Logic*. Cambridge University Press.

Börger, E.; Grädel, E.; and Gurevich, Y. 1997. *The Classical Decision Problem*. Springer-Verlag.

Chandra, A.; Kozen, D.; and Stockmeyer, L. 1981. Alternation. *JACM* 28.

Finger, M., and Gabbay, D. 1992. Adding a temporal dimension to a logic system. *JoLLI* 2.

Gabbay, D.; Kurusz, A.; Wolter, F.; and Zakharyaschev, M. 2003. *Many-dimensional Modal Logics: Theory and Applications*. Elsevier.

Lutz, C.; Wolter, F.; and Zakharyaschev, M. 2008. Temporal description logics: A survey. In *Proc. TIME'08*, IEEE Press.

Lutz, C. 2001. Interval-based temporal reasoning with general TBoxes. In *Proc. IJCAI'01*, AAAI Press.

Pnueli, A. 1977. The temporal logic of programs. In *Proc. FOCS'77*.

Schild, K. 1993. Combining terminological logics with tense logic. In *Proc. EPIA'93*, Springer LNCS 727.

Schmidt-Schauß, M., and Smolka, G. 1991. Attributive concept descriptions with complements. *AIJ* 48.

Schmiedel, A. 1990. A temporal terminological logic. In *Proc. AAAI'90*, AAAI Press.

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)*.

Tobies, S. 1999. A NEXPTIME-complete description logic strictly contained in C^2. In *Proc. CSL'99*, Springer LNCS 1683.

Wolter, F., and Zakharyaschev, M. 1999. Temporalizing description logics. In *Proc. FroCoS'98*, Wiley.

Appendix

In this appendix, we show the hardness part of Theorem 9. Note that the proof of the hardness part of Theorem 11 is simpler than the proof given below. It can be found in (Baader, Ghilardi, & Lutz 2008).

Lemma 20. *Satisfiability in $\mathcal{ALC}$-LTL$|_{gGCI}$ w.r.t. rigid names is* 2-EXPTIME*-hard.*

Proof. The proof is by reduction of the word problem for exponentially space bounded alternating Turing machines (ATMs). An ATM is of the form $\mathcal{M} = (Q, \Sigma, \Gamma, q_0, \Theta)$, where $Q = Q_\exists \uplus Q_\forall \uplus \{q_a, q_r\}$ is a finite set of *states*, partitioned into *existential states* from $Q_\exists$, *universal states* from $Q_\forall$, an *accepting state* q_a, and a *rejecting state* q_r; Σ is the *input alphabet* and $\Gamma \supseteq \Sigma$ the *work alphabet* containing a *blank symbol* $B \notin \Sigma$; $q_0 \in Q_\exists \cup Q_\forall$ is the *initial* state; and the *transition relation* Θ is of the form $\Theta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R\}$. We write $\Theta(q, \sigma)$ for $\{(q', \theta, M) \mid (q, \sigma, q', b, M) \in \Theta\}$.

A *configuration* of an ATM is a word wqw' with $w, w' \in \Gamma^*$ and $q \in Q$. The intended meaning is that the (one-sided infinite) tape contains the word ww' with only blanks behind it, the machine is in state q, and the head is on the left-most symbol of w'. The *successor configurations* of a configuration wqw' are defined in the usual way in terms of the transition relation Θ. A *halting configuration* is of the form wqw' with $q \in \{q_a, q_r\}$. We may assume w.l.o.g. that any configuration other than a halting configuration has at least one successor configuration. A *computation* of an ATM $\mathcal{M}$ on a word w is a (finite or infinite) sequence of successive configurations $K_1, K_2, \ldots$. The ATMs considered here have only finite computations on any input. Since this case is simpler than the general one, we only define acceptance for ATMs with finite computations and refer to (Chandra, Kozen, & Stockmeyer 1981) for the full definition. Let $\mathcal{M}$ be such an ATM. A halting configuration is *accepting* iff it is of the form wq_aw'. For other configurations $K = wqw'$, the acceptance behaviour depends on q: if $q \in Q_\exists$, then K is accepting iff at least one successor configuration is accepting; if $q \in Q_\forall$, then K is accepting iff all successor configurations are accepting. Finally, the ATM $\mathcal{M}$ with initial state q_0 *accepts* the input w iff the *initial configuration* q_0w is accepting. We use $L(\mathcal{M})$ to denote the language accepted by $\mathcal{M}$, i.e.,

$$L(\mathcal{M}) = \{w \in \Sigma^* \mid \mathcal{M} \text{ accepts } w\}.$$

The *word problem* for $\mathcal{M}$ is the following decision problem: given a word $w \in \Sigma^*$, does $w \in L(\mathcal{M})$ hold or not?

There exists an exponentially space bounded ATM $\mathcal{M} = (Q, \Sigma, \Gamma, q_0, \Theta)$ whose word problem is 2-EXPTIME-hard (Chandra, Kozen, & Stockmeyer 1981). Our aim is to reduce the word problem for this ATM $\mathcal{M}$ to satisfiability in $\mathcal{ALC}$-LTL$|_{gGCI}$ w.r.t. rigid names. We may assume that the length of every computation of $\mathcal{M}$ on $w \in \Sigma^k$ is bounded by 2^{2^k}, and all the configurations wqw' in such computations satisfy $|ww'| \leq 2^k$. We may also assume w.l.o.g. that $\mathcal{M}$ never attempts to move to the left when it is on the left-most tape cell.

Let $w = \sigma_0 \cdots \sigma_{k-1} \in \Sigma^*$ be an input to $\mathcal{M}$. We construct an $\mathcal{ALC}$-LTL$|_{gGCI}$ formula $\phi_{\mathcal{M},w}$ such that $w \in L(\mathcal{M})$ iff $\phi_{\mathcal{M},w}$ is satisfiable w.r.t. rigid names. In an $\mathcal{ALC}$-LTL structure satisfying $\phi_{\mathcal{M},w}$, each domain element from Δ describes a single tape cell of a configuration of $\mathcal{M}$. The formula $\phi_{\mathcal{M},w}$, that will be defined below is actually not of the syntactic form $\Box\mathcal{B} \wedge \varphi$ (where $\mathcal{B}$ is a conjunction of $\mathcal{ALC}$-axioms and φ is an $\mathcal{ALC}$-LTL formula that does not contain GCIs) required for $\mathcal{ALC}$-LTL formulae with global GCIs. Instead, $\phi_{\mathcal{M},w}$ is a conjunction of formulae of the form

- $\Box\alpha$ where α is an $\mathcal{ALC}$-axiom,
- ψ where ψ is an $\mathcal{ALC}$-LTL formula not containing GCIs.

Since $\Box$ distributes over conjunction, it is obvious that such a formula is equivalent to an $\mathcal{ALC}$-LTL formula with global GCIs. In the definition of $\phi_{\mathcal{M},w}$, we use the following symbols:

- a single individual name a that identifies the first tape cell of the first configuration;
- a single *rigid* role name r to represent "going to the next tape cell in the same configuration" and "going from the last tape cell in a configuration to the first tape cell in a successor configuration";
- the elements of Q and Γ are viewed as *rigid* concept names;
- *rigid* concept names $A_0, \ldots, A_{k-1}$ are the bits of a binary counter that numbers the tape cells in each configuration;
- auxiliary *rigid* concept name I and H; I indicates the initial configuration and H indicates that, in the current configuration, the head is to the left of the current tape cell;
- auxiliary *rigid* concept names $T_{q,\sigma,M}$ for all $q \in Q$, $\sigma \in \Gamma$, and $M \in \{L, R\}$; intuitively, $T_{q,\sigma,M}$ is true if, in the current configuration, the head is on the left neighboring cell and the machine executes transition (q, σ, M);
- for each element of Q and Γ, a flexible concept name which is distinguished from its rigid version by a prime;
- (flexible) concept names $A'_0, \ldots, A'_{k-1}$ to realize an orthogonal counter (in the sense that it counts along the temporal dimension instead of along r).

Before giving the formal reduction, let us explain the underlying intuition. As said above, a single configuration is described as a sequence of r-successors of length 2^k of the individual representing its first tape cell. The tape cells of a configuration are numbered from 0 to $2^k - 1$, using the counter realized through the concept names $A_0, \ldots, A_{k-1}$. We denote the concept that expresses that the counter has value $i, 0 \leq i < 2^k$, by $(C_A = i)$; i.e., $(C_A = 0)$ denotes $\neg A_0 \sqcap \neg A_1 \sqcap \ldots \sqcap \neg A_{k-1}$, $(C_A = 1)$ denotes $A_0 \sqcap \neg A_1 \sqcap \ldots \sqcap \neg A_{k-1}$, ..., $(C_A = 2^k - 1)$ denotes $A_0 \sqcap A_1 \sqcap \ldots \sqcap A_{k-1}$.

The r-successor of the last tape cell of a given configuration represents the first tape cell of a successor configuration of this configuration. It obtains the number 0, i.e., the counter realized by $A_0, \ldots, A_{k-1}$ is reset to 0, which simply means that we count modulo 2^k. Since we have an *alternating* Turing machine, it is not enough to consider one sequence of configurations. For a configuration with a universal state, we must consider all successor configurations. Thus, we do not consider a single sequence of r-successors, but rather a tree of r-successors.

The main problem to solve when defining the reduction is to ensure that each configuration following a given configuration in the tree of r-successors is actually a successor configuration, i.e., tape cells that are not immediately to the left or right of the head remain unchanged, and the other tape cells are changed according to the transition relation. For the first type of cells this means that, given a cell numbered i in the current configuration, the next cell with the same number should carry the same symbol. However, we cannot remember the value i of the A-counter when going down along the sequence of r-successors since this counter is incremented (modulo 2^k) when going to an r-successor. This is where the temporal dimension comes into play. Here, we realize an A'-counter, using the (flexible) concept names $A'_0, \ldots, A'_{k-1}$, whose value does not change along the r-dimension, but is incremented (modulo 2^k) along the temporal dimension. This additional counter, together with the flexible copies of the symbols from Q and Γ, can be used to transfer a symbol from a tape cell in a given configuration to the corresponding tape cell in a successor configuration (see below).

In the following, we use $\phi \rightarrow \psi$ as an abbreviation for $\neg\phi \vee \psi$, $C \Rightarrow D$ as an abbreviation for $\neg C \sqcup D$, and $C \Leftrightarrow D$ as an abbreviation for $(C \Rightarrow D) \sqcap (D \Rightarrow C)$.

The reduction formula $\phi_{\mathcal{M},w}$ is the conjunction of the following formulae:

We start by setting up I, H, r, and the A-counter:

- I behaves as described, i.e., it marks the initial configuration, whose first tape cell is represented by the individual a:

$$\Box\,(a : I)$$
$$\Box\,\big(I \sqcap \neg(C_A = 2^k - 1) \sqsubseteq \forall r.I\big)$$

- H behaves as described, i.e., it marks the tape cells that are to the right of the head, where the head position is indicated by having a state concept at this cell:

$$\Box\left((H \sqcup \bigsqcup_{q \in Q} q) \sqcap \neg(C_A = 2^k - 1) \sqsubseteq \forall r.H\right)$$

- there is always an r-successor, except when we meet the head in a halting configuration:

$$\Box\,(\neg(q_a \sqcup q_r) \sqsubseteq \exists r.\top)$$

- the counter realized by $A_0, \ldots, A_{k-1}$ has value 0 at a, and it is incremented along r (modulo 2^k):

$$\Box\,(a : (C_A = 0))$$
$$\Box\Big(\top \sqsubseteq \bigsqcap_{i<k}\big(\bigsqcap_{j<i} A_j\big) \Rightarrow \big((A_i \Rightarrow \forall r.\neg A_i) \sqcap (\neg A_i \Rightarrow \forall r.A_i)\big)\Big)$$
$$\Box\Big(\top \sqsubseteq \bigsqcap_{i<k}\big(\bigsqcup_{j<i} \neg A_j\big) \Rightarrow \big((A_i \Rightarrow \forall r.A_i) \sqcap (\neg A_i \Rightarrow \forall r.\neg A_i)\big)\Big)$$

Some properties of runs of ATMs can be formalized without using the temporal dimension:

- The initial configuration is the one induced by the input $w = \sigma_0 \dots \sigma_{k-1}$:

$$\begin{aligned} &\Box\,(a : \forall r^i.\sigma_i) && \text{for } i < k \\ &\Box\,(a : \forall r^k.B) \\ \Box\big(&I \sqcap B \sqcap \neg(C_A = 2^k - 1) \sqsubseteq \forall r.B\big) \end{aligned}$$

- The computation starts on the left-most tape cell of this initial configuration in state q_0:

$$\Box\,(a : q_0)$$

- Each tape cell is labelled with exactly one symbol and at most one state:

$$\Box\left(\top \sqsubseteq \bigsqcup_{\sigma\in\Gamma}(\sigma \sqcap \neg \bigsqcap_{\theta\in\Gamma\setminus\{\sigma\}} \neg\theta)\right)$$
$$\Box\left(\top \sqsubseteq \bigsqcap_{p,q\in Q, p\neq q} \neg(p \sqcap q)\right)$$

- There is only one head position per configuration:

$$\Box\left(H \sqsubseteq \bigsqcap_{q\in Q} \neg q\right)$$

It remains to implement the transitions and to say that symbols not under the head do not change in successor configurations. Here we need the temporal dimension. We start with setting up the A'-counter:

- for every value of the A'-counter realized using the (flexible) concept names $A'_0, \dots, A'_{k-1}$, there is a time point at which a has that value:

$$\Box\Big(\bigwedge_{i<k}\big(\bigwedge_{j<i} a : A'_j\big) \rightarrow \big((a : A'_i \rightarrow \mathsf{X}a : \neg A'_i) \wedge (a : \neg A'_i \rightarrow \mathsf{X}a : A'_i)\big)\Big)$$
$$\Box\Big(\bigwedge_{i<k}\big(\bigvee_{j<i} a : \neg A'_j\big) \rightarrow \big((a : A'_i \rightarrow \mathsf{X}a : A'_i) \wedge (a : \neg A'_i \rightarrow \mathsf{X}a : \neg A'_i)\big)\Big)$$

  This is basically the same formula as for the A-counter, but the values of the A'-counter are considered for the fixed initial individual a, and they are incremented along the temporal dimension.

- The value of the A'-counter is preserved along r, i.e., for all $i, 0 \leq i < k$, we require:

$$\Box\,(A'_i \sqsubseteq \forall r.A'_i)$$
$$\Box\,(\neg A'_i \sqsubseteq \forall r.\neg A'_i)$$

In summary, we have associated one "temporal slice" with each counter value of the second counter. In the following, we use $(C_A = C_{A'})$ to denote the concept $(A_0 \Leftrightarrow A'_0) \sqcap \dots \sqcap (A_{k-1} \Leftrightarrow A'_{k-1})$, which states that the value of the A-counter coincides with the value of the A'-counter. Accordingly, $(C_A = C_{A'} + 1 \bmod 2^k)$ expresses that the value of the A-counter is equal to the value of the A'-counter plus 1 (modulo 2^k), which can be expressed by a recasting of the incrementation concept given already twice above:

$$\bigsqcap_{i<k}\big(\bigsqcap_{j<i} A'_j\big) \Rightarrow \big((A'_i \Rightarrow \neg A_i) \sqcap (\neg A'_i \Rightarrow A_i)\big) \sqcap$$
$$\bigsqcap_{i<k}\big(\bigsqcup_{j<i} \neg A'_j\big) \Rightarrow \big((A'_i \Rightarrow A_i) \sqcap (\neg A'_i \Rightarrow \neg A_i)\big)$$

The concept $(C_A = C_{A'} + 2 \bmod 2^k)$, which expresses that the value of the A-counter is equal to the value of the A'-counter plus 2 (modulo 2^k), can be defined similarly, using an auxiliary set $A''_0, \dots, A''_{k-1}$ of flexible concept names.

- We can now say that symbols not under the head do not change:

$$\begin{aligned} &\Box\left(\sigma \sqcap \bigsqcap_{q\in Q} \neg q \sqcap (C_A = C_{A'}) \sqsubseteq \forall r.\sigma'\right) && \text{for all } \sigma \in \Gamma \\ &\Box\,(\sigma' \sqcap \neg(C_A = C_{A'}) \sqsubseteq \forall r.\sigma') && \text{for all } \sigma \in \Gamma \\ &\Box\,(\sigma' \sqcap (C_A = C_{A'}) \sqsubseteq \sigma) && \text{for all } \sigma \in \Gamma \end{aligned}$$

- Transitions are implemented in a similar way. The fact that we have an *alternating* Turing is taken into account by enforcing a branching on universal transitions:

$$\Box\left(q \sqcap \sigma \sqsubseteq \bigsqcup_{(p,\nu,M)\in\Theta(q,\sigma)} \forall r.T_{p,\nu,M}\right) \quad \text{for all } q \in Q_\exists, \sigma \in \Sigma$$

$$\Box\left(q \sqcap \sigma \sqsubseteq \bigsqcap_{(p,\nu,M)\in\Theta(q,\sigma)} \exists r.T_{p,\nu,M}\right) \quad \text{for all } q \in Q_\forall, \sigma \in \Sigma$$

$$\Box\,\big(T_{q,\sigma,M} \sqcap (C_A = C_{A'} + 1 \bmod 2^k) \sqsubseteq \forall r.\sigma'\big) \quad \text{for all } \sigma \in \Gamma, q \in Q, M \in \{L, R\}$$

$$\Box\,(T_{q,\sigma,R} \sqcap (C_A = C_{A'}) \sqsubseteq \forall r.q') \quad \text{for all } \sigma \in \Gamma, q \in Q$$

$$\Box\,\big(T_{q,\sigma,L} \sqcap (C_A = C_{A'} + 2 \bmod 2^k) \sqsubseteq \forall r.q'\big) \quad \text{for all } \sigma \in \Gamma, q \in Q$$

$$\begin{aligned} &\Box\,(q' \sqcap \neg(C_A = C_{A'}) \sqsubseteq \forall r.q') && \text{for all } q \in Q \\ &\Box\,(q' \sqcap (C_A = C_{A'}) \sqsubseteq q) && \text{for all } q \in Q \end{aligned}$$

It remains to encode the fact that the input $w = \sigma_0 \dots \sigma_{k-1}$ is accepted. Since any computation of $\mathcal{M}$ is terminating, and halting configurations (i.e., configurations with state q_a or q_r) are the only ones without successor configurations, this can be done as follows:

- We can express the fact that the initial configuration for input w is accepting by disallowing the state q_r to occur:

$$\Box\,(\top \sqsubseteq \neg q_r)$$

This finishes the definition of $\phi_{\mathcal{M},w}$, which is the conjunction of the formulae introduced above. It is easy to see that the size of $\phi_{\mathcal{M},w}$ is polynomial in k, and that $\phi_{\mathcal{M},w}$ is satisfiable w.r.t. rigid names iff $w \in L(\mathcal{M})$. ❏