Metric Temporal Description Logics with Interval-Rigid Names

FRANZ BAADER, Institute of Theoretical Computer Science, Technische Universität Dresden, Germany
STEFAN BORGWARDT, Institute of Theoretical Computer Science, Technische Universität Dresden, Germany
PATRICK KOOPMANN, Institute of Theoretical Computer Science, Technische Universität Dresden, Germany
ANA OZAKI, Department of Informatics, University of Bergen, Norway and KRDB Research Centre, Free University of Bozen-Bolzano, Italy
VERONIKA THOST, MIT-IBM Watson AI Lab, IBM Research, USA

In contrast to qualitative linear temporal logics, which can be used to state that some property will eventually be satisfied, metric temporal logics allow us to formulate constraints on how long it may take until the property is satisfied. While most of the work on combining description logics (DLs) with temporal logics has concentrated on qualitative temporal logics, there is a growing interest in extending this work to the quantitative case. In this paper, we complement existing results on the combination of DLs with metric temporal logics by introducing *interval-rigid* concept and role names. Elements included in an interval-rigid concept or role name are required to stay in it for some specified amount of time. We investigate several combinations of (metric) temporal logics with $\mathcal{ALC}$ by either allowing temporal operators only on the level of axioms, or also applying them to concepts. In contrast to most existing work on the topic, we consider a timeline based on the integers and also allow assertional axioms. We show that the worst-case complexity does not increase beyond the previously known bound of 2-ExpSpace, and investigate in detail how this complexity can be reduced by restricting the temporal logic and the occurrences of interval-rigid names.

CCS Concepts: • **Theory of computation → Description logics**; *Modal and temporal logics*; • **Computing methodologies → Description logics**; *Temporal reasoning*.

Additional Key Words and Phrases: temporal description logics, metric temporal logics, interval-rigid names

ACM Reference Format:
Franz Baader, Stefan Borgwardt, Patrick Koopmann, Ana Ozaki, and Veronika Thost. 2020. Metric Temporal Description Logics with Interval-Rigid Names. *ACM Trans. Comput. Logic* 21, 4, Article 30 (August 2020), 45 pages. https://doi.org/10.1145/3399443

Authors' addresses: Franz Baader, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, franz.baader@tu-dresden.de; Stefan Borgwardt, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, stefan.borgwardt@tu-dresden.de; Patrick Koopmann, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, patrick.koopmann@tu-dresden.de; Ana Ozaki, Department of Informatics, University of Bergen, Bergen, Norway, KRDB Research Centre, Free University of Bozen-Bolzano, Bozen-Bolzano, Italy, ana.ozaki@uib.no; Veronika Thost, MIT-IBM Watson AI Lab, IBM Research, Cambridge, MA, USA, veronika.thost@ibm.com.

1529-3785/2020/8-ART30 $15.00
https://doi.org/10.1145/3399443

1 INTRODUCTION

Description logics (DLs) [19] are a well-investigated family of logic-based knowledge representation languages, which provide the formal basis for the Web Ontology Language OWL.[1] As a consequence, DL-based ontologies are employed in many application areas, and they are particularly successful in the medical domain (see, e.g., the medical ontologies Galen and SNOMED CT[2]). For example, the concept of a patient with a concussion can formally be expressed in DLs with the *concept (expression)* $\texttt{Patient} \sqcap \exists \texttt{finding}.\texttt{Concussion}$, which is built from the *concept names* (i.e., unary predicates) `Patient` and `Concussion` and the *role name* (i.e., binary predicate) `finding` using the *concept constructors* conjunction ($\sqcap$) and existential restriction ($\exists$). Concepts and roles are used within terminological and assertional axioms to state facts about the application domain, such as that concussion is a disease ($\texttt{Concussion} \sqsubseteq \texttt{Disease}$) and that patient Bob has a concussion ($\texttt{Patient}(\texttt{BOB}), \exists \texttt{finding}.\texttt{Concussion}(\texttt{BOB})$).

This example, given by Baader et al. [20], also illustrates a shortcoming of classical DLs. For a doctor, it is important to know whether the concussed patient has lost consciousness, which is the reason why SNOMED CT contains a concept for "concussion with no loss of consciousness" [44]. However, the temporal pattern inherent in this concept (after the concussion, the patient remained conscious until the examination) cannot be modelled in classical DLs such as in the one used in SNOMED CT.

This problem of classical DLs has been generally recognised, and a great variety of temporal extensions of DLs have been investigated in the literature.[3] In this paper, we focus on the DL $\mathcal{ALC}$ [43] and metric variants of the temporal logic LTL [41], a point-based temporal logic whose semantics is based on a linear flow of time represented by the set of integers $\mathbb{Z}$. But, even with these two logics being fixed, there are still several other design decisions to be made. For instance, the temporal operators can be applied to axioms [20] and/or inside axioms, i.e., to concepts [30, 48] and to roles [39]. The former, for example, allows us to state that Bob has not lost consciousness *since* ($\mathcal{S}$) he had a concussion:

$$(\exists \texttt{finding}.\texttt{Conscious}(\texttt{BOB}))\, \mathcal{S}\, (\exists \texttt{finding}.\texttt{Concussion}(\texttt{BOB})).$$

The latter allows us to formalise "concussion with no loss of consciousness" independently of a specific individual as a (temporal) concept

$$(\exists \texttt{finding}.\texttt{Conscious})\, \mathcal{S}\, (\exists \texttt{finding}.\texttt{Concussion}).$$

Another decision to be made is whether *rigid concepts and roles* are considered. In contrast to flexible concept and role names, whose interpretation can be different at different time points, the interpretation of rigid names does not change over time. Obviously, it makes sense to consider concepts like `Human` and roles like `hasFather` as rigid, but `Conscious` and `finding` as flexible (i.e., not rigid). If the logic allows temporal operators within concepts, then rigid concepts can be expressed using terminological axioms, but rigid roles cannot. In fact, the latter usually render the combined logic undecidable [39]. In contrast, in the setting considered by Baader et al. [20], rigid roles do not cause undecidability, but adding rigidity leads to an increase in complexity.

In this paper, we address a shortcoming of the purely qualitative temporal DLs mentioned above. The qualitative $\mathcal{S}$-operator in our example does not say anything about how long after the concussion the examination happened. However, the above definition of "concussion with no loss of consciousness" is only sensible in case the examination took place shortly after the concussion. Otherwise, an intermediate loss of consciousness could also have been due to other causes. Another

[1] https://www.w3.org/TR/owl2-overview/

[2] see http://www.opengalen.org/ and http://www.snomed.org/

[3] We refer the reader to [30, 39] for an overview of the field of temporal DLs.

use case where quantitative temporal patterns must be modelled in the medical domain is the formulation of eligibility criteria in clinical trials [15, 27]. For example, one may want to describe patients who have a reaction caused by a treatment between 45 and 180 days ago, and who had no additional treatment since then, which could be done with the following concept:

$$(\exists \texttt{finding.Reaction}) \sqcap \big((\neg \exists \texttt{procedure.Treatment})\, \mathcal{S}_{[45,180]} (\exists \texttt{procedure.Treatment})\big).$$

On the temporal logic side, extensions of LTL by such intervals have been investigated in detail [2, 3, 38]. They can actually be simulated in qualitative LTL using the temporal operator $\bigcirc$ (see Section 3.2). However, if the interval boundaries are encoded in binary, this leads to an exponential blowup. The complexity results by Alur and Henzinger [3] imply that this blowup cannot in general be avoided. On the other hand, Lutz et al. [38] show that using intervals of a restricted form (where the lower bound is 0) does not increase the complexity compared to the qualitative case. The combination of the DL $\mathcal{ALC}$ with a metric extension of LTL was first investigated by Gutiérrez-Basulto et al. [32]. That paper considers both the case where temporal operators are applied only within concepts and the case where they are applied both within concepts and outside of terminological axioms. Our research extends and thus complements these results by considering also *assertional knowledge*, *past operators with the timeline of integers* $\mathbb{Z}$, and, most importantly, *interval-rigid names*.

By allowing also for assertional axioms, our language can describe the temporal behaviour of specific individuals, as in the example above. We show that temporal operators on assertional axioms come at no additional cost. In most cases, assertions can be encoded using temporal concepts or temporal terminological axioms.

Since domain knowledge often refers to both past and future, we include corresponding temporal operators and accordingly define the semantics over the integers $\mathbb{Z}$ rather than over the natural numbers $\mathbb{N}$. It is known that, over $\mathbb{N}$, past operators offer no additional expressivity, but can express temporal properties exponentially more succinctly [37]. As some of our lower bounds also apply to $\mathbb{N}$, our results are therefore also interesting for this semantics. Defining the semantics over $\mathbb{Z}$ is further motivated by our novel concept of interval-rigid names discussed next, which allows us to refer to the past even without using past operators.

Interval-rigid names are a recently introduced means of expressiveness [16]. They can be seen as a metric variant of rigid names and thus fit into our setting of metric temporal DLs. In a nutshell, they can be used to express that individuals belonging to a concept need to belong to that concept for at least k consecutive time points, and similarly for roles. For example, one can encode the usual duration of diseases, e.g., that an influenza infection (usually) lasts at least one week, by making the role `influenzaFinding` rigid for 7 days.

The outline of this paper is as follows. In Section 2, we recall the constructors of $\mathcal{ALC}$ and LTL with metric temporal operators, and the combined logic $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$. In Section 3, we show that previous results [32, 39, 48] also hold in our slightly different setting: we additionally consider assertional axioms and employ a temporal semantics over the timeline $\mathbb{Z}$ instead of $\mathbb{N}$. In Section 4, we further extend the logic $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ with *interval-rigid* names. We show that the worst-case complexity of 2-ExpSpace does not increase, and can be reduced to 2-ExpTime under an additional assumption (see Table 2). In Section 5, we consider the effect of adding both interval-rigid concepts and roles as well as metric temporal operators to the logic $\mathcal{ALC}$-LTL [20], where temporal operators can only be applied to axioms. Interestingly, in the presence of rigid roles, interval-rigid concepts actually cause undecidability. Without rigid roles, the addition of interval-rigid concepts and roles leaves the logic decidable, but matches the 2-ExpSpace bound mentioned above, surprisingly even when metric temporal operators are disallowed. Finally, in Section 6, we investigate the complexity of this logic without interval-rigid names. Essentially, this extends the analysis of Baader et al. [20] to quantitative temporal operators.

This paper is based on the conference paper [16]. Our main contributions are as follows.

- We provide a detailed complexity analysis of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ and various fragments with and without (interval-)rigid names (see Tables 1 and 2 for an overview). This comes mainly from [16], but we adapt these results here for a temporal semantics based on $\mathbb{Z}$ instead of $\mathbb{N}$.
- We include a new 2-ExpSpace-hardness result (Theorem 5.4).
- We develop two reductions for encoding assertions as terminological axioms, thereby simplifying many constructions (Theorems 3.1 and 3.2).
- We describe small extensions of existing complexity results for $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ and some fragments, to deal with integers and assertions (Theorem 3.7).

To provide a better reading experience, the more technical details of our proofs are given in the appendix.

2 THE TEMPORAL DESCRIPTION LOGIC $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$

We first introduce the DL $\mathcal{ALC}$ and its metric temporal extension $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ [32], which augments $\mathcal{ALC}$ by allowing metric temporal logic operators [2] both within $\mathcal{ALC}$ axioms and to combine these axioms. We actually consider a slight extension of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ (as it was proposed originally) by assertional axioms, and consider a timeline based on $\mathbb{Z}$ instead of $\mathbb{N}$. In Section 3, we show that this does not change the complexity of reasoning compared to existing results [32].

Syntax. Let $\mathsf{N_C}$, $\mathsf{N_R}$, and $\mathsf{N_I}$ be countably infinite sets of *concept names*, *role names*, and *individual names*, respectively. An $\mathcal{ALC}$ *concept* is an expression defined by

$$C, D ::= A \mid \top \mid \neg C \mid C \sqcap D \mid \exists r.C,$$

where $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$.

$\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *concepts* extend $\mathcal{ALC}$ concepts with the *temporal concept constructors* $\bigcirc C$, $\bigcirc^- C$, $C\,\mathcal{U}_I D$, and $C\,\mathcal{S}_I D$,where I is an interval of the form $[c_1, c_2]$ or $[c_1, \infty)$ with $c_1, c_2 \in \mathbb{N}$, $c_1 \leq c_2$, given in *binary*. We may use $[c_1, c_2)$ to abbreviate $[c_1, c_2 - 1]$, and similarly for the left endpoint.

An $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *axiom* is either a *general concept inclusion (GCI)* of the form $C \sqsubseteq D$ or an *assertion* of the form $C(a)$ or $r(a, b)$, where C, D are $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ concepts, $r \in \mathsf{N_R}$, and $a, b \in \mathsf{N_I}$. $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae* are expressions of the form

$$\varphi, \psi ::= \alpha \mid \top \mid \neg\varphi \mid \varphi \wedge \psi \mid \bigcirc\varphi \mid \bigcirc^-\varphi \mid \varphi\,\mathcal{U}_I\psi \mid \varphi\,\mathcal{S}_I\psi,$$

where α is an $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ axiom.

Semantics. Description logics semantics is as in first-order logic, meaning that the symbols in $\mathsf{N_C}$, $\mathsf{N_R}$, and $\mathsf{N_I}$, are interpreted as unary and binary predicates, and constants, respectively. A DL *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ over a non-empty set $\Delta^{\mathcal{I}}$, called the *domain*, defines an *interpretation function* $\cdot^{\mathcal{I}}$ that maps each concept name $A \in \mathsf{N_C}$ to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name $r \in \mathsf{N_R}$ to a binary relation $r^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$ and each individual name $a \in \mathsf{N_I}$ to an element $a^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$. As usual, we extend the mapping $\cdot^{\mathcal{I}}$ from concept names to $\mathcal{ALC}$ concepts as follows.

$$\top^{\mathcal{I}} := \Delta^{\mathcal{I}} \qquad (\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \qquad (C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$$
$$(\exists r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid \text{ there is } e \in C^{\mathcal{I}} \text{ such that } (d, e) \in r^{\mathcal{I}}\}$$

The temporal semantics we consider here is based on the timeline $\mathbb{Z}$. A *(temporal DL) interpretation* is a structure $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$, where each $\mathcal{I}_i = (\Delta^{\mathfrak{I}}, \cdot^{\mathcal{I}_i})$, $i \in \mathbb{Z}$, is a DL interpretation over $\Delta^{\mathfrak{I}}$ (*constant domain assumption*) and $a^{\mathcal{I}_i} = a^{\mathcal{I}_j}$ for all $a \in \mathsf{N_I}$ and $i, j \in \mathbb{Z}$, i.e., the interpretation of

individual names is fixed. The mappings $\cdot^{\mathcal{I}_i}$ are extended to $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ concepts as follows.

$$
\begin{aligned}
(\bigcirc C)^{\mathcal{I}_i} &:= C^{\mathcal{I}_{i+1}} \\
(\bigcirc^- C)^{\mathcal{I}_i} &:= C^{\mathcal{I}_{i-1}} \\
(C\,\mathcal{U}_I D)^{\mathcal{I}_i} &:= \{d \in \Delta^{\mathfrak{I}} \mid \text{there exists } k \text{ with } k - i \in I,\ d \in D^{\mathcal{I}_k},\ \text{and } d \in C^{\mathcal{I}_j} \text{ for all } j \in [i,k)\} \\
(C\,\mathcal{S}_I D)^{\mathcal{I}_i} &:= \{d \in \Delta^{\mathfrak{I}} \mid \text{there exists } k \text{ with } i - k \in I,\ d \in D^{\mathcal{I}_k},\ \text{and } d \in C^{\mathcal{I}_j} \text{ for all } j \in (k,i]\}
\end{aligned}
$$

That is, the concept $C\,\mathcal{U}_I D$ consists of all elements that satisfy D at some point k in the interval I relative to the current time point i, and satisfy C at all time points between i and k (including i itself). For example, $\exists r.\bigcirc A \sqcap A\,\mathcal{U}_{[2,5)} B$ is an $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ concept that describes those domain elements that have an r-successor that will satisfy the concept name A at the next time point, and themselves satisfy A until, after at least 2 and at most 4 time points in the future, they will satisfy B.

Validity of an $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ formula φ in $\mathfrak{I}$ at $i \in \mathbb{Z}$ (written $\mathfrak{I}, i \models \varphi$) is inductively defined as follows.

$$
\begin{aligned}
&\mathfrak{I}, i \models C \sqsubseteq D && \text{iff} && C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}. \\
&\mathfrak{I}, i \models C(a) && \text{iff} && a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}. \\
&\mathfrak{I}, i \models r(a,b) && \text{iff} && (a^{\mathcal{I}_i}, b^{\mathcal{I}_i}) \in r^{\mathcal{I}_i}. \\
&\mathfrak{I}, i \models \neg\varphi && \text{iff} && \mathfrak{I}, i \not\models \varphi. \\
&\mathfrak{I}, i \models \varphi \wedge \psi && \text{iff} && \mathfrak{I}, i \models \varphi \text{ and } \mathfrak{I}, i \models \psi. \\
&\mathfrak{I}, i \models \bigcirc\varphi && \text{iff} && \mathfrak{I}, i+1 \models \varphi. \\
&\mathfrak{I}, i \models \bigcirc^-\varphi && \text{iff} && \mathfrak{I}, i-1 \models \varphi. \\
&\mathfrak{I}, i \models \varphi\,\mathcal{U}_I \psi && \text{iff} && \text{there is } k \text{ with } k - i \in I,\ \mathfrak{I}, k \models \psi \text{ and } \mathfrak{I}, j \models \varphi \text{ for all } j \in [i,k). \\
&\mathfrak{I}, i \models \varphi\,\mathcal{S}_I \psi && \text{iff} && \text{there is } k \text{ with } i - k \in I,\ \mathfrak{I}, k \models \psi,\ \text{and } \mathfrak{I}, j \models \varphi \text{ for all } j \in (k,i].
\end{aligned}
$$

As usual, we define the abbreviations $\bot := \neg\top$, $C \sqcup D := \neg(\neg C \sqcap \neg D)$, $C \to D := \neg C \sqcup D$, $C \leftrightarrow D := (C \to D) \sqcap (D \to C)$, $C \equiv D := (C \sqsubseteq D) \wedge (D \sqsubseteq C)$, $\forall r.C := \neg(\exists r.\neg C)$, $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$, $\varphi \to \psi := \neg\varphi \vee \psi$, $\varphi \leftrightarrow \psi := (\varphi \to \psi) \wedge (\psi \to \varphi)$, $\Diamond_I \alpha := \top\,\mathcal{U}_I \alpha$, and $\Box_I \alpha := \neg\Diamond_I \neg\alpha$, and the past operators $\Diamond_I^- \alpha := \top\,\mathcal{S}_I \alpha$, $\Box_I^- \alpha := \neg\Diamond_I^- \neg\alpha$, $\Diamond^- \alpha := \top\,\mathcal{S}\alpha$, and $\Box^- \alpha := \neg\Diamond^- \neg\alpha$, where C, D are concepts, φ, ψ formulae and α, β are either concepts or formulae [19, 30]. Furthermore, we may omit the interval I if it is of the form $[0,\infty)$, e.g. write $\alpha\,\mathcal{U}\beta$ instead of $\alpha\,\mathcal{U}_{[0,\infty)}\beta$. Given the semantics of $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$, $\bigcirc\alpha$ is equivalent to $\Diamond_{[1,1]}\alpha$ and $\Box_{[1,1]}\alpha$, and $\bigcirc^-\alpha$ is equivalent to $\Diamond^-_{[1,1]}\alpha$ and $\Box^-_{[1,1]}\alpha$. However, we cannot use this to simulate $\bigcirc\alpha$ and $\bigcirc^-\alpha$ in all the logics that we investigate.

Given a formula φ, we denote by $\mathsf{N_C}(\varphi)$ the set of all concept names occurring in φ, and similarly for $\mathsf{N_R}(\varphi)$ and $\mathsf{N_I}(\varphi)$. We denote by $\text{sub}^{\text{c}}(\varphi)$ the set of all subconcepts occurring in φ, and by $\text{sub}^{\text{f}}(\varphi)$ its subformulae. Finally, $\text{cl}^{\text{c}}(\varphi)$ is the closure of

$$\text{sub}^{\text{c}}(\varphi) \cup \{C\,\mathcal{U}D \mid C\,\mathcal{U}_{[c,\infty)}D \in \text{sub}^{\text{c}}(\varphi)\} \cup \{C\,\mathcal{S}D \mid C\,\mathcal{S}_{[c,\infty)}D \in \text{sub}^{\text{c}}(\varphi)\}$$

under single negation, and likewise for $\text{cl}^{\text{f}}(\varphi)$ and $\text{sub}^{\text{f}}(\varphi)$.

Sequences. We use the following notation for sequences. We define a sequence of objects indexed by integers as a function $\sigma\colon X \to Y$, where X is a (possibly infinite) interval in $\mathbb{Z}$, and Y is a set of objects (e.g., DL interpretations). We use the abbreviation $\sigma^{\leq i}$ for the subsequence of σ defined on $X \cap (-\infty, i]$, and similarly for $\sigma^{>i}$, $\sigma^{<i}$, etc. Assuming that $[i,j] \subseteq X$, we write $\sigma^{[i,j]}$ for the finite subsequence $\sigma(i)\,\sigma(i+1)\ldots\sigma(j)$. The concatenation of two finite sequences σ_1, σ_2 is denoted by $\sigma_1\sigma_2$. Furthermore, we define $\sigma^1 := \sigma$, $\sigma^{n+1} := (\sigma^n)\sigma$, and $\sigma^\omega := \sigma\sigma\ldots$, and correspondingly for the other direction ${}^\omega\sigma := \ldots\sigma\sigma$.

Reasoning. We are interested in the complexity of the *satisfiability* problem in $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$, i.e., of deciding whether there exists an interpretation $\mathfrak{I}$ such that $\mathfrak{I}, 0 \models \varphi$ holds for a given $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$

Table 1. Overview over the features of the temporal description logics we investigate.

	Temporal concepts	Temporal assertions	Temporal GCIs	Metric operators
$\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$	✓	✓	✓	✓
$\text{LTL}_{\mathcal{ALC}}$	✓	✓	✓	–
$\text{LTL}^{\text{bin}}_{\mathcal{ALC}\mid gGCI}$	✓	✓	–	✓
$\text{LTL}_{\mathcal{ALC}\mid gGCI}$	✓	✓	–	–
$\mathcal{ALC}$-LTL^{bin}	–	✓	✓	✓
$\mathcal{ALC}$-LTL	–	✓	✓	–
$\mathcal{ALC}$-$\text{LTL}^{\text{bin}}_{\mid gGCI}$	–	✓	–	✓
$\mathcal{ALC}$-$\text{LTL}_{\mid gGCI}$	–	✓	–	–

formula φ. There are applications in which only the state of individuals changes over time, while the meaning of concepts remains the same throughout the timeline. In these applications, it is sufficient to use temporal operators only at the level of assertions and concepts, but not at the level of GCIs, which can lead to a decrease in complexity.

To account for this, we also consider a syntactic restriction from [20]: we say that φ is an $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ *formula with global GCIs* if it is of the form $\Box\Box^{-}\mathcal{T} \wedge \varphi$, where $\mathcal{T}$ is a conjunction of GCIs and φ is an $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ formula without GCIs. Note that φ may be formulated with arbitrary combinations of assertions using Boolean and temporal operators. Following the notation in [20], we denote the resulting logic by $\text{LTL}^{\text{bin}}_{\mathcal{ALC}\mid gGCI}$.

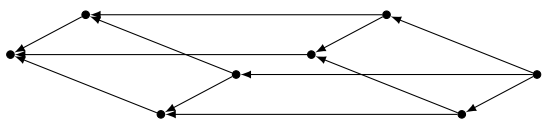
Fragments. Throughout the paper, we discuss several fragments of the logic $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$. An overview of all these fragments is given in Table 1. Here, by *temporal concepts* we mean that the logic allows temporal operators on concepts, by *temporal assertions* that the logic supports temporal operators on assertions, by *temporal GCIs* that the logic supports temporal operators on GCIs (rather than global GCIs only), and by *metric operators* that intervals other than $[0, \infty)$ may be used in the temporal operators. For instance, $\mathcal{ALC}$-LTL supports only the qualitative temporal operators $\bigcirc$, $\bigcirc^{-}$, $\mathcal{U}$, and $\mathcal{S}$ on GCIs and assertions, but not on concepts. The logic designated as $\text{LTL}^{\text{bin}}_{\mathcal{ALC}\mid gGCI}$ allows temporal operators $\mathcal{U}_I$ and $\mathcal{S}_I$ on concepts and assertions, but not on GCIs, which means that only global GCIs are supported. Table 2 summarises our new complexity results, as well as known complexity results, for the fragments considered. The sets of rigid names and interval-rigid names, as described in the introduction, are denoted by N_{Rig} and N_{IRig}, respectively, and will be introduced in detail in Section 4. For example, $\mathsf{N}_{\text{Rig}} = \emptyset$ describes the case when there are no rigid names, $\mathsf{N}_{\text{Rig}} \subseteq \mathsf{N}_{\text{C}}$ means that only concept names can be rigid, and the case $\mathsf{N}_{\text{Rig}} \subseteq \mathsf{N}_{\text{C}} \cup \mathsf{N}_{\text{R}}$ includes also rigid roles.

3 FIRST RESULTS

Before we introduce and discuss interval-rigid names, we lift several existing results to the setting we consider in this paper. Specifically, our syntax includes assertional axioms in formulae and the semantics is defined over the integers. In the literature on temporal DLs, assertions and the semantics over $\mathbb{Z}$ have already been considered [8, 20, 48]. To the best of our knowledge, the latter has not been investigated in the context of $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ or its fragment $\text{LTL}_{\mathcal{ALC}}$.

In Section 3.1, we first show that reasoning about $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ formulae can be polynomially reduced to reasoning about $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ formulae as proposed originally, without assertions. This allows us to

Table 2. Overview of all complexity results. In the second column, each logic is denoted by a dot, and inclusions between logics are drawn as arrows. All complexity results except "in 2-ExpTime" are tight. Below the results, "$\leq$" denotes the references for the upper bounds, and "$\geq$" the lower bounds.

<table>
<tr><th rowspan="2">Logic</th><th rowspan="2">Inclusions</th><th colspan="2">$N_{Rig} \subseteq N_C \cup N_R$</th><th colspan="2">$N_{Rig} \subseteq N_C$ or $N_{Rig} = \emptyset$</th><th colspan="2">$N_{Rig} \subseteq N_C$</th><th>$N_{Rig} = \emptyset$</th></tr>
<tr><th>$N_{IRig} \subseteq N_C$</th><th>$N_{IRig} = \emptyset$</th><th>$N_{IRig} \subseteq N_C \cup N_R$</th><th>$N_{IRig} \subseteq N_C$</th><th>$N_{IRig} = \emptyset$</th><th>$N_{IRig} = \emptyset$</th></tr>
<tr><td>$\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$</td><td rowspan="8">  </td><td rowspan="8">undecidable
Thm. 5.5</td><td rowspan="4">undecidable
[39]</td><td rowspan="2">2-ExpSpace
$\leq$ Thm. 4.4</td><td rowspan="2">2-ExpSpace
$\geq$ [32], Thm. 3.7</td><td rowspan="2">2-ExpSpace
$\geq$ [32], Thm. 3.7</td><td rowspan="2">$\leq$ [48], Thm. 3.7, $\geq$ [30, 39]</td></tr>
<tr><td>$\mathrm{LTL}_{\mathcal{ALC}}$</td></tr>
<tr><td>$\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$</td><td rowspan="2">2-ExpTime
$\leq$ Thm. 4.6
$\geq$ Thm. 4.7</td><td rowspan="2">ExpSpace
$\leq$ [32], Thm. 3.7</td></tr>
<tr><td>$\mathrm{LTL}_{\mathcal{ALC}|gGCI}$</td><td>ExpTime
$\leq$ [39], Thm. 3.7</td></tr>
<tr><td>$\mathcal{ALC}$-$\mathrm{LTL}^{\mathrm{bin}}$</td><td rowspan="4">2-ExpTime
$\leq$ Thm. 6.2
$\geq$ [20]</td><td rowspan="4">2-ExpSpace
$\geq$ Thm. 5.4</td><td rowspan="2">ExpSpace
$\geq$ Thm. 5.2</td><td rowspan="2">ExpSpace
$\leq$ Thm. 5.3</td></tr>
<tr><td>$\mathcal{ALC}$-LTL</td></tr>
<tr><td>$\mathcal{ALC}$-$\mathrm{LTL}^{\mathrm{bin}}_{|gGCI}$</td><td rowspan="2">in 2-ExpTime</td><td>NExpTime
[20]</td><td>ExpTime
$\leq$ [20]</td></tr>
<tr><td>$\mathcal{ALC}$-$\mathrm{LTL}_{|gGCI}$</td><td>ExpSpace
$\geq$ [3, 29]</td><td>ExpTime
$\geq$ [43]</td></tr>
</table>

simplify the presentation in the rest of the paper by disregarding assertions. Second, in Section 3.2 we consider the sublanguage $\mathrm{LTL}_{\mathcal{ALC}}$ of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ that allows only the qualitative $\mathcal{U}$- and $\mathcal{S}$-operators (without intervals), as well as the intermediate language $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$, where intervals are only of the forms $[0, c]$ or $[c, \infty)$, $c \in \mathbb{N}$. While $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ is exponentially more succinct than $\mathrm{LTL}_{\mathcal{ALC}}$ in general, reasoning in $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ can be *polynomially* reduced to reasoning in $\mathrm{LTL}_{\mathcal{ALC}}$. Thus, all complexity results for $\mathrm{LTL}_{\mathcal{ALC}}$ also apply to $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$, and we do not need to consider the latter in subsequent sections. In Section 3.3, we extend known complexity results for $\mathrm{LTL}_{\mathcal{ALC}}$ and $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ over the natural numbers $\mathbb{N}$ to the integer timeline we consider here.

3.1 Encoding Assertions into GCIs

It turns out that for some of the settings considered in this paper, we can simulate assertions using GCIs. However, we need to use temporal operators on concepts, and hence this result does not apply to the logics we study in Sections 5 and 6.

Global GCIs. We first consider the case of an $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$ formula $\varphi \wedge \Box\Box^{-}\mathcal{T}$, where φ contains no GCIs and $\mathcal{T}$ is a conjunction of GCIs. The idea is to dispose of φ entirely, by encoding the evolution of the named individuals as it is enforced by φ into the evolution of a single individual:

- we represent relevant formulae ψ over assertions using fresh concept names A_ψ,
- we simulate their semantics using additional GCIs, and
- we enforce the existence of an element that satisfies A_φ.

For every role assertion $r(a, b) \in \mathsf{sub}^{\mathsf{f}}(\varphi)$, we introduce a fresh concept name $A_{r(a,b)}$; for every $C \in \mathsf{sub}^{\mathsf{c}}(\varphi \wedge \mathcal{T})$ and $a \in \mathsf{N_I}(\varphi)$, a fresh concept name $A_{C(a)}$; and, for every $r \in \mathsf{N_R}(\varphi)$ and $a \in \mathsf{N_I}(\varphi)$, a fresh role r_a. The roles r_a are used to relate anonymous r-successors originally required for a to our representative individual. The semantics of the concept assertions $C(a)$ with $C \in \mathsf{sub}^{\mathsf{c}}(\varphi \wedge \mathcal{T})$ and $a \in \mathsf{N_I}(\varphi)$ is then encoded using the following definitions for the concept names $A_{C(a)}$.

(a) $A_{\top(a)} \equiv \top$
(b) $A_{(\neg C)(a)} \equiv \neg A_{C(a)}$
(c) $A_{(C \sqcap D)(a)} \equiv A_{C(a)} \sqcap A_{D(a)}$
(d) $A_{(\exists r.C)(a)} \equiv \exists r_a.C \sqcup \bigsqcup_{b \in \mathsf{N_I}(\varphi)} (A_{r(a,b)} \sqcap A_{C(b)})$
(e) $A_{(\bigcirc C)(a)} \equiv \bigcirc A_{C(a)}$
(f) $A_{(\bigcirc^{-} C)(a)} \equiv \bigcirc^{-} A_{C(a)}$
(g) $A_{(C\,\mathcal{U}_I D)(a)} \equiv A_{C(a)}\,\mathcal{U}_I A_{D(a)}$
(h) $A_{(C\,\mathcal{S}_I D)(a)} \equiv A_{C(a)}\,\mathcal{S}_I A_{D(a)}$

Additionally, we add the GCI $A_{C(a)} \sqsubseteq A_{D(a)}$ for every $C \sqsubseteq D \in \mathcal{T}$ and $a \in \mathsf{N_I}(\varphi)$. To simulate the rest of φ, we similarly use fresh concept names A_ψ for all $\psi \in \mathsf{sub}^{\mathsf{f}}(\varphi)$, with the following definitions.

- $A_\top \equiv \top$
- $A_{\neg\psi} \equiv \neg A_\psi$
- $A_{\psi_1 \wedge \psi_2} \equiv A_{\psi_1} \sqcap A_{\psi_2}$
- $A_{\bigcirc\psi} \equiv \bigcirc A_\psi$
- $A_{\bigcirc^{-}\psi} \equiv \bigcirc^{-} A_\psi$
- $A_{\psi_1\,\mathcal{U}_I\,\psi_2} \equiv A_{\psi_1}\,\mathcal{U}_I A_{\psi_2}$
- $A_{\psi_1\,\mathcal{S}_I\,\psi_2} \equiv A_{\psi_1}\,\mathcal{S}_I A_{\psi_2}$

Denote by $\mathcal{T}'$ the conjunction of $\mathcal{T}$ with all the above GCIs. Then we can show that $\varphi \wedge \Box\Box^{-}\mathcal{T}$ is satisfiable iff $\Box\Box^{-}(\mathcal{T}' \wedge (\top \sqsubseteq \exists r_0.A_\varphi))$ is satisfiable, where r_0 is a fresh role name.

Theorem 3.1. *Satisfiability of* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$ *formulae can be polynomially reduced to satisfiability of* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$ *formulae without assertions.*

Note that $\Box\Box^-(\mathcal{T}' \wedge (\top \sqsubseteq \exists r_0.A_\varphi))$ is an $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$-formula with only global GCIs and without assertions. However, since the semantics of both (temporal) subformulae and assertions are encoded with the help of (temporal) concepts in GCIs, this construction may introduce temporal concept operators, even if they are not present in the original formula.

Local GCIs. We next investigate the more general setting and consider φ to be an arbitrary $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ formula. In contrast to the above, we now encode the temporal assertions into *every* individual by means of the concept names A_α as before. As the GCIs to be satisfied are now changing over time, we cannot use a direct encoding of each GCI $C \sqsubseteq D$ into $A_{C(a)} \sqsubseteq A_{D(a)}$ to ensure that the local GCIs are also satisfied by the named individuals. Instead, we use the formula

$$\Box\Box^-\Big((C \sqsubseteq D) \to \bigwedge_{a \in \mathsf{N_I}(\varphi)} (A_{C(a)} \sqsubseteq A_{D(a)})\Big)$$

for every $C \sqsubseteq D$ occurring in φ to ensure this, i.e., whenever a GCI is satisfied, it also needs to be satisfied for all (implicitly encoded) individual names. To ensure that all domain elements always agree on the interpretation of the concept names A_α, we use the formulae

$$\Box\Box^-\big((\top \equiv A_\alpha) \vee (\bot \equiv A_\alpha)\big)$$

for all assertions α we consider. We denote the conjunction of all the above formulae by $\varphi_{\mathcal{A}}$ and again define a set $\mathcal{T}'$ of global GCIs that contains the axioms defined by Items (a)–(h) above. Finally, the formula φ' is obtained from φ by replacing every assertion α in φ by $\top \equiv A_\alpha$. Then φ is satisfiable iff $\varphi' \wedge \varphi_{\mathcal{A}} \wedge \Box\Box^-\mathcal{T}'$ is satisfiable.

Theorem 3.2. *Satisfiability of* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ *formulae can be polynomially reduced to satisfiability of* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ *formulae without assertions.*

3.2 Relation of $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ and $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ to $\mathrm{LTL}_{\mathcal{ALC}}$

We now discuss the relations between $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ and two of its fragments, $\mathrm{LTL}_{\mathcal{ALC}}$ and $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$. Recall that the notation $\cdot^{\mathrm{bin}}$ refers to the fact that the endpoints of the intervals are given in binary. This does not increase the expressivity in comparison to $\mathrm{LTL}_{\mathcal{ALC}}$ [39], where only the qualitative $\mathcal{U}$- and $\mathcal{S}$-operators are allowed. In fact, one can expand any formula $\varphi\,\mathcal{U}_{[c_1,c_2]}\psi$ to

$$\bigvee_{c_1 \le i \le c_2} \Big(\bigcirc^i \psi \wedge \bigwedge_{0 \le j < i} \bigcirc^j \varphi\Big),$$

where $\bigcirc^i$ denotes i nested $\bigcirc$-operators. Likewise, $\varphi\,\mathcal{U}_{[c_1,\infty)}\psi$ is equivalent to

$$\Big(\bigwedge_{0 \le i < c_1} \bigcirc^i \varphi\Big) \wedge \bigcirc^{c_1}(\varphi\,\mathcal{U}\psi).$$

Corresponding equivalences hold for concepts and formulae containing the operator $\mathcal{S}_I$. However, if this transformation is recursively applied to all subexpressions, then the size of the resulting formula is exponential: ignoring the nested $\bigcirc$ and $\bigcirc^-$ operators, its syntax tree has polynomial depth and an exponential branching factor; and the $\bigcirc^i$- and $(\bigcirc^-)^i$ formulae have exponential depth, but introduce no branching. This blowup cannot be avoided, because satisfiability in $\mathrm{LTL}_{\mathcal{ALC}}$ is ExpSpace-complete, while for $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ it is 2-ExpSpace-complete [32]. This gap exists even for propositional temporal logics, where the complexity jumps from PSpace to ExpSpace when going from LTL to $\mathrm{LTL}^{\mathrm{bin}}$ [3].

Proposition 3.3. *Every* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formula can be translated in exponential time into an equivalent* $\mathrm{LTL}_{\mathcal{ALC}}$ *formula.*

Observe that, if this transformation is applied only to formulae (not to concepts), then the axioms and concepts occurring in the formula remain the same, but they may occur more often afterwards. Similarly, if the transformation is applied only to concepts, then the number of subformulae remains the same, but the axioms and concepts occurring in them may be larger. Nevertheless, it is to be expected that reasoning in $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ gets harder than in $\mathrm{LTL}_{\mathcal{ALC}}$ by an exponential factor. As mentioned above, this is not the case for $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$, where all intervals must be of the form $[0, c]$ or $[c, \infty)$, as we show next.

Previously, it has been shown for a branching temporal logic that $\mathcal{U}_{[0,c]}$ can be simulated by the classical $\mathcal{U}$-operator, while the size of the formula is only increased by a polynomial factor [38]. The basic idea is to use, for each formula $\varphi\,\mathcal{U}_{[0,c]}\psi$, a counter that determines the temporal distance to the *nearest* occurrence of ψ that makes $\varphi\,\mathcal{U}\psi$ true (i.e., ψ is satisfied at that time point, and φ is satisfied at all time points in between). The formula $\varphi\,\mathcal{U}_{[0,c]}\psi$ is then satisfied iff the counter value does not exceed c. Note that for each time point satisfying $\varphi\,\mathcal{U}\psi$, there is always a unique nearest time point satisfying ψ, which uniquely determines the counter value. To satisfy $\varphi\,\mathcal{U}_{[0,c]}\psi$, only counter values below c are relevant, which is why it is sufficient to encode the counter using n concept names, where n is the number of bits in the binary representation of c.

To extend this result to formulae of the form $\varphi\,\mathcal{U}_{[c,\infty)}\psi$, we use a similar counter, which however determines the distance to the *furthest* occurrence of ψ that makes $\varphi\,\mathcal{U}\psi$ true. Then, $\varphi\,\mathcal{U}_{[c,\infty)}\psi$ is satisfied iff the counter value is at least c. Again, only counter values below c are relevant to detect the satisfaction of $\varphi\,\mathcal{U}_{[c,\infty)}\psi$: if the counter value is higher, the particular value is not relevant to determine whether $\varphi\,\mathcal{U}_{[c,\infty)}\psi$ is satisfied, we only need to remember that we reached this counter value and that φ remained satisfied. Consequently, we can represent the counter again using only polynomially many bits. These ideas can be straightforwardly extended to past operators and temporal concepts. The proof can be found in the appendix.

Theorem 3.4. *Every* $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ *formula can be translated in polynomial time into an equisatisfiable* $\mathrm{LTL}_{\mathcal{ALC}}$ *formula.*

Similar to Proposition 3.3, this reduction preserves certain properties of the original formula: if the $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ formula contains only global GCIs or contains no temporal concepts, then so does the resulting $\mathrm{LTL}_{\mathcal{ALC}}$ formula. In fact, the reduction applies to all sublogics of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ that we consider in this paper. Hence, in the remainder of this paper we do not explicitly consider logics with the superscript $\cdot^{0,\infty}$, because they have the same complexity as the corresponding temporal DLs that use only $\mathcal{U}$ and $\mathcal{S}$.

3.3 Satisfiability in $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ and $\mathrm{LTL}_{\mathcal{ALC}}$

The complexity of concept satisfiability in $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ with respect to global *TBoxes* (corresponding to $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$ formulae without assertions) and of satisfiability of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *temporal TBoxes* (i.e., $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formulae without assertions) has been investigated previously for semantics defined over $\mathbb{N}$ [32, 39, 48]. We reformulate these results in our setting, that is, for a semantics over $\mathbb{Z}$ and for a syntax including past operators. By Theorems 3.1 and 3.2, these results apply also to formulae with assertions.

Our constructions are not substantially different to those in [32], and provide mainly an adaptation for temporal past operations and to the semantics over $\mathbb{Z}$. We repeat them here in full mainly in preparation of our main results in Section 4. Specifically, we introduce well-known proof ideas that are based on the notion of *quasimodels* [32, 39, 48]. Quasimodels are abstractions of interpretations

in which each time point is represented by a set of *types*, called a *quasistate*. Each type describes the interpretation for a single domain element at some time point, while a quasistate collects the information about all domain elements at some time point. It is central for the complexity results that every satisfiable formula has a quasimodel of a certain restricted form that, in particular, can be guessed and verified using only exponential space. We define quasimodels [32, 39, 48] for the $\mathbb{Z}$ timeline, and then show that this approach also yields ExpSpace- and ExpTime-decision procedures in our setting.

Let φ be an $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ formula or an $\mathrm{LTL}_{\mathcal{ALC}}$ formula without assertions. A *concept type for* φ is any subset t of $\mathrm{cl}^{\mathrm{c}}(\varphi)$ such that

T1 $\neg C \in t$ iff $C \notin t$, for all $\neg C \in \mathrm{cl}^{\mathrm{c}}(\varphi)$, and
T2 $C \sqcap D \in t$ iff $C, D \in t$, for all $C \sqcap D \in \mathrm{cl}^{\mathrm{c}}(\varphi)$.

Intuitively, concept types are used to model the correct behaviour of the basic concept constructors $\neg$ and $\sqcap$. Similarly, we define *formula types* t as subsets of $\mathrm{cl}^{\mathrm{f}}(\varphi)$ satisfying the following conditions:

T1' $\neg\alpha \in t$ iff $\alpha \notin t$, for all $\neg\alpha \in \mathrm{cl}^{\mathrm{f}}(\varphi)$, and
T2' $\alpha \wedge \beta \in t$ iff $\alpha, \beta \in t$, for all $\alpha \wedge \beta \in \mathrm{cl}^{\mathrm{f}}(\varphi)$.

A *quasistate* for φ is a set Q of concept and formula types such that

S1 Q contains exactly one formula type, denoted by t_Q,
S2 if $t \in Q$ and $\exists s.D \in t$, then there is $t' \in Q$ with $\{D\} \cup \{\neg E \mid \neg\exists s.E \in t\} \subseteq t'$, and
S3 for all $C \sqsubseteq D \in \mathrm{cl}^{\mathrm{f}}(\varphi)$, we have $C \sqsubseteq D \in t_Q$ iff $C \in t$ implies $D \in t$ for all concept types $t \in Q$.

A quasistate represents a single interpretation via the concept types of its domain elements, in addition to a formula type that specifies the axioms satisfied by this interpretation. In this way, the seven conditions above deal with the non-temporal part of the semantics of $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$.

To model the temporal dimension, in the following we consider sequences of types that are infinite in both directions, representing the evolution of domain elements (via concept types) and axioms (via formula types). We say that a (finite or infinite) sequence r of types *realises an until expression* $\alpha\,\mathcal{U}_I\beta$ *at position* n if there is a k with $k-n \in I$ with $\beta \in r(k)$ and $\alpha \in r(j)$ for all $j \in [n, k)$. Similarly, a (finite or infinite) sequence r of types *realises a since expression* $\alpha\,\mathcal{S}_I\beta$ *at position* n if there is a k with $n-k \in I$ with $\beta \in r(k)$ and $\alpha \in r(j)$ for all $j \in (k, n]$. We may omit 'at position n' if $n = 0$.

The evolution over the complete timeline is described by *concept (formula) runs* for φ, which are sequences $\ldots r(-1)\,r(0)\,r(1)\ldots$ of concept (formula) types that are infinite in both directions and satisfy the following conditions for all $n \in \mathbb{Z}$, where cl^* is cl^{c} for concept runs and cl^{f} for formula runs:

R1 for all $\bigcirc\alpha \in \mathrm{cl}^*(\varphi)$, $\bigcirc\alpha \in r(n)$ iff $\alpha \in r(n+1)$,
R2 for all $\bigcirc^-\alpha \in \mathrm{cl}^*(\varphi)$, $\bigcirc^-\alpha \in r(n)$ iff $\alpha \in r(n-1)$,
R3 for all $\alpha\,\mathcal{U}_I\beta \in \mathrm{cl}^*(\varphi)$, $\alpha\,\mathcal{U}_I\beta \in r(n)$ iff r realises $\alpha\,\mathcal{U}_I\beta$ at position n, and
R4 for all $\alpha\,\mathcal{S}_I\beta \in \mathrm{cl}^*(\varphi)$, $\alpha\,\mathcal{S}_I\beta \in r(n)$ iff r realises $\alpha\,\mathcal{S}_I\beta$ at position n.

It now only remains to combine quasistates and runs into a coherent representation of a temporal model. A *quasimodel* for φ is a pair $(S, \mathfrak{R})$, where S is a mapping from $\mathbb{Z}$ to the set of all quasistates and $\mathfrak{R}$ is a possibly infinite set of runs such that

M1 $\varphi \in t_{S(0)}$,
M2 for all $r \in \mathfrak{R}$ and $n \in \mathbb{Z}$, $r(n) \in S(n)$, and
M3 for all $n \in \mathbb{Z}$ and $t \in S(n)$, there is a run $r \in \mathfrak{R}$ such that $r(n) = t$.

Intuitively, **M1** expresses that φ is satisfied at time point 0. By **M2** and **M3**, types on a run are in fact elements of a quasistate at the corresponding position and, conversely, every type in a

quasistate occurs in some run (at the corresponding time point). By Condition **S1**, $\mathfrak{R}$ contains exactly one formula run.

We now establish a central result for showing the subsequent complexity upper bounds. In short, to decide satisfiability it suffices to look for a quasimodel of a certain regular shape. The main difference to previous results [32, 39, 48] is that we need this regularity to be present in both directions of the timeline given by $\mathbb{Z}$.

Lemma 3.5. *An* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ *formula* φ *is satisfiable iff it has a quasimodel* $(S, \mathfrak{R})$ *in which* S *is of the form*

$$^{\omega}\big(S(-k_1)\dots S(-k_2-1)\big)\, S(-k_2)\dots S(0)\dots S(k_3)\, \big(S(k_3+1)\dots S(k_4)\big)^{\omega},$$

where k_1, k_2, k_3 *and* k_4 *are bounded double exponentially in the size of* φ.

Consider now an $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ formula $\varphi = \psi \wedge \Box\Box^{-}\mathcal{T}$ with global GCIs, i.e., where ψ contains only assertions and $\mathcal{T}$ is a conjunction of GCIs. Recall that, by Theorem 3.1, we can disregard assertions entirely, so that we may assume w.l.o.g. that φ is of the form $\Box\Box^{-}\mathcal{T}$, with $\mathcal{T}$ containing only GCIs. If φ is of that form, we only need one formula type, namely the one that contains $\Box\Box^{-}\mathcal{T}$, $\Box^{-}\mathcal{T}$, and all subformulae of $\mathcal{T}$. As a consequence, the formula run maps every time point to the same formula type. We call such a formula run *constant*. With a constant formula run, satisfiability coincides with the existence of a *constant quasimodel*: a quasimodel $(S, \mathfrak{R})$ where $S(i) = S(j)$ for all $i, j \in \mathbb{Z}$; that is, all quasistates are the same.

Lemma 3.6. *An* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$ *formula* $\Box\Box^{-}\mathcal{T}$ *is satisfiable iff it has a constant quasimodel.*

Note, however, that a regular behaviour as in Lemma 3.5 also shows up in the concept runs, which represent the evolution of domain elements.

Theorem 3.7 summarises the complexity of the logics $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ and $\mathrm{LTL}_{\mathcal{ALC}}$ and their fragments with only global GCIs. It turns out that their respective complexity remains the same, regardless of whether we consider $\mathbb{Z}$ or $\mathbb{N}$ for the temporal semantics.

Theorem 3.7. *The satisfiability problem is 2-ExpSpace-complete in* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$, *ExpSpace-complete in* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$ *and* $\mathrm{LTL}_{\mathcal{ALC}}$ *and ExpTime-complete in* $\mathrm{LTL}_{\mathcal{ALC}|gGCI}$.

Proof sketch. Membership in 2-ExpSpace is a consequence of a result presented below, in Theorem 4.4. The lower bound is a consequence of a result in [32, Theorem 5], which shows 2-ExpSpace-hardness for $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ over the natural numbers. To see that it transfers to our case, note that we can easily reduce satisfiability of $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ formulae without past operators over the natural numbers to satisfiability over the integers. Indeed, without past operators, φ is satisfiable over $\mathbb{N}$ iff it is satisfiable over $\mathbb{Z}$. The exact same argument can be used to transfer the ExpSpace lower bound for $\mathrm{LTL}_{\mathcal{ALC}}$ using the semantics over the natural numbers [30, 39] to our case. The other lower bounds follow from the fact that satisfiability is ExpSpace-hard for $\mathrm{LTL}^{\mathrm{bin}}$ [3, 29] and ExpTime-hard for $\mathcal{ALC}$ [43]. The remaining upper bounds follow from the our results on the shape of quasimodels; details are provided in the appendix. □

4 $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ WITH INTERVAL-RIGID NAMES

Many temporal DLs allow so-called *rigid* names, whose interpretation is not allowed to change over time. To formally define this notion, we fix a countable set $\mathsf{N}_{\mathsf{Rig}} \subseteq \mathsf{N}_{\mathsf{C}} \cup \mathsf{N}_{\mathsf{R}}$ of infinitely many *rigid* concept names and infinitely many *rigid* role names, and require interpretations $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ to *respect* these names, in the sense that $X^{\mathcal{I}_i} = X^{\mathcal{I}_j}$ should hold for all $X \in \mathsf{N}_{\mathsf{Rig}}$ and $i, j \in \mathbb{Z}$. Note that $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ can make concepts effectively rigid using (global) axioms of the form $A \equiv \bigcirc A$. That is, rigid concepts are just "syntactic sugar". On the other hand, rigid roles lead to undecidability even

in $\mathrm{LTL}_{\mathcal{ALC}}$ [39]. For these reasons, for $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$, we do not get interesting results by considering rigid names explicitly. However, they will become meaningful in the later sections, where we look at other logics.

To augment the expressivity of temporal DLs while avoiding undecidability, we propose *interval-rigid* names. In contrast to rigid names, interval-rigid names only need to remain rigid for a limited period of time. Formally, we fix another countably infinite set $\mathsf{N}_{\mathsf{IRig}} \subseteq (\mathsf{N}_{\mathsf{C}} \cup \mathsf{N}_{\mathsf{R}}) \setminus \mathsf{N}_{\mathsf{Rig}}$ of *interval-rigid names*, and a function $\mathsf{iRig} \colon \mathsf{N}_{\mathsf{IRig}} \to \mathbb{N}$ such that, for each $n \in \mathbb{N}$, there are infinitely many concept names A with $\mathsf{iRig}(A) = n$ and infinitely many role names r with $\mathsf{iRig}(r) = n$. An interpretation $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i \in \mathbb{Z}})$ *respects* the interval-rigid names if the following holds for all $X \in \mathsf{N}_{\mathsf{IRig}}$ and $i \in \mathbb{Z}$, where $\mathsf{iRig}(X) = k$.

$$\text{For each } d \in X^{\mathcal{I}_i}, \text{ there is a } j \in \mathbb{Z} \text{ such that } i \in [j, j+k) \text{ and } d \in X^{\mathcal{I}_\ell} \text{ for all } \ell \in [j, j+k).$$

Intuitively, any element (or pair of elements) d in the interpretation of an interval-rigid name must be in that interpretation for at least k consecutive time points. We call such a name *k-rigid*. The names in $(\mathsf{N}_{\mathsf{C}} \cup \mathsf{N}_{\mathsf{R}}) \setminus (\mathsf{N}_{\mathsf{Rig}} \cup \mathsf{N}_{\mathsf{IRig}})$ are called *flexible*, and we assume that there are still infinitely many flexible concept names as well as infinitely many flexible role names. For convenience, we implicitly consider the function iRig to assign 1 to all flexible names, which does not affect the semantics. Using this convention, we do not have to distinguish between flexible names and interval-rigid names. For the complexity analysis, in the following we consider as input not only an $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formula φ, but additionally the restriction $\mathsf{iRig}|_\varphi$ of the function iRig to those interval-rigid names that occur in φ. We consider the *size* of this function to be the size of the largest number in its codomain (in binary representation).

Example 4.1. To illustrate the effect of interval-rigid names, we make the example given in the introduction more explicit. We might make the role name `influenzaFinding` rigid for 7 days to express that an influenza infection lasts at least one week: $\mathsf{iRig}(\texttt{influenzaFinding}) = 7$. Consider the following simplified $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formula, containing a GCI expressing that someone with an influenza finding is ill, that `patient1` was known to have been healthy for a week 3 days ago, and that he was diagnosed with influenza yesterday:

$$\begin{aligned}\varphi = {} & \Box\Box^{-}(\exists \texttt{influenzaFinding.Finding} \sqsubseteq \neg \texttt{HealthyPatient}) \\ & \wedge \Box_{[-3,-10]} \texttt{HealthyPatient}(\texttt{patient1}) \\ & \wedge \bigcirc^{-} \texttt{influenzaFinding}(\texttt{patient1}, \texttt{finding1}).\end{aligned}$$

Due to the interval-rigidity of `influenzaFinding`, this formula implies that the patient will continue to have influenza for at least 4 days:

$$\varphi \models \Box_{[0,4]} \texttt{influenzaFinding}(\texttt{patient1}, \texttt{finding1}).$$

We investigate the complexity of *satisfiability with (interval-)rigid names*, which is defined as before, but considers only interpretations that respect (interval-)rigid names. The decision problem of *satisfiability with (interval-)rigid concepts* is defined correspondingly, with the restriction that $\mathsf{N}_{\mathsf{Rig}} \subseteq \mathsf{N}_{\mathsf{C}}$ ($\mathsf{N}_{\mathsf{IRig}} \subseteq \mathsf{N}_{\mathsf{C}}$). Note that (interval-)rigid roles can be used to simulate (interval-)rigid concepts via existential restrictions $\exists r.\top$ [20]. Therefore, the case where only role names can be (interval-)rigid is not easier than the general case.

Interval-rigid *concepts* $A \in \mathsf{N}_{\mathsf{C}} \cap \mathsf{N}_{\mathsf{IRig}}$, $\mathsf{iRig}(A) = k$, can be simulated by global $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ GCIs of the form

$$\Box\Box^{-}\big(\neg A \sqsubseteq \bigcirc(A \to \Box_{[0,k)} A)\big).$$

Thus, Theorem 3.7 directly yields the upper bounds for this case (see Table 2). For the sublogics of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ that we investigate later, this is not always so easy.

The complexity of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ with interval-rigid *roles* is harder to establish. We first show in Section 4.1 the membership in 2-ExpSpace, by extending the well-known quasimodel construction to interval-rigid names. 2-ExpSpace-hardness holds already for the case without interval-rigid names or assertions [32] (see also Theorem 3.7). In Section 4.2, we then show 2-ExpTime-completeness for the fragment with global GCIs. To simplify the proofs of the upper bounds, in this section we assume without loss of generality that

- $\mathsf{N}_{\mathsf{Rig}} = \emptyset$ since rigid concepts can be simulated and rigid roles make satisfiability undecidable,
- $\mathsf{N}_{\mathsf{IRig}} \subseteq \mathsf{N}_{\mathsf{R}}$ since interval-rigid concepts can be simulated as described above (actually we treat all roles as interval-rigid by using the above convention that $\mathsf{iRig}(r) = 1$ for all flexible roles r), and
- formulae do not contain assertions (see Theorems 3.1 and 3.2 and the following lemma).

Lemma 4.2. *In the presence of interval-rigid names, satisfiability of* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae (with global GCIs) can be polynomially reduced to satisfiability of* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae (with global GCIs and) without assertions.*

4.1 Satisfiability is in 2-ExpSpace

For the 2-ExpSpace upper bound, we extend the quasimodel approach [32, 39] of Section 3. Recall that quasistates contain types describing the interpretation of the domain elements at one time point. To capture interval-rigid role relations, we consider quasistates that cover the temporal evolution of domain elements over a window of fixed width. Satisfiability is still characterised by the existence of regular quasimodels. We take the notions from Section 3 and extend them to the new setting. The definitions of concept and formula types remain unchanged.

Let φ be an $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formula. To put an upper bound on the time window we have to look at, we consider the largest number occurring in φ and $\mathsf{iRig}|_\varphi$ and denote it by ℓ_φ. For infinite concept and formula runs, we now consider subsequences of length $2\ell_\varphi + 1$ to account for the influence of the interval-rigid names on the temporal evolution of domain elements and axioms, respectively. Formally, a *concept (formula) run segment* for φ is defined as a sequence $\sigma(-\ell_\varphi) \dots \sigma(0) \dots \sigma(\ell_\varphi)$ composed exclusively of concept (formula) types, respectively, such that

R1 for all $\bigcirc\alpha \in \mathsf{cl}^*(\varphi)$, $\bigcirc\alpha \in \sigma(0)$ iff $\alpha \in \sigma(1)$,
R2 for all $\bigcirc^-\alpha \in \mathsf{cl}^*(\varphi)$, $\bigcirc^-\alpha \in \sigma(0)$ iff $\alpha \in \sigma(-1)$,
R3 for all $\alpha\,\mathcal{U}_I\beta \in \mathsf{cl}^*(\varphi)$, we have $\alpha\,\mathcal{U}_I\beta \in \sigma(0)$ iff either (a) σ realises $\alpha\,\mathcal{U}_I\beta$ at 0 or (b) I is of the form $[c, \infty)$ and $\alpha, \alpha\,\mathcal{U}\beta \in \sigma(i)$ for all $i \in [0, \ell_\varphi]$, and
R4 for all $\alpha\,\mathcal{S}_I\beta \in \mathsf{cl}^*(\varphi)$, we have $\alpha\,\mathcal{S}_I\beta \in \sigma(0)$ iff either (a) σ realises $\alpha\,\mathcal{S}_I\beta$ at 0 or (b) I is of the form $[c, \infty)$ and $\alpha, \alpha\,\mathcal{S}\beta \in \sigma(i)$ for all $i \in [-\ell_\varphi, 0]$,

where cl^* is either cl^{c} or cl^{f} (as appropriate). In contrast to Section 3.3, Conditions **R1** and **R2** now evaluate the semantics of $\bigcirc$ and $\bigcirc^-$ only at time point 0, because run segments will be composed later into full runs in a quasimodel. Conditions **R3** and **R4** deal with the semantics of $\mathcal{U}$ and $\mathcal{S}$, but only up to the finite horizon of ℓ_φ and $-\ell_\varphi$, respectively.

We accordingly redefine *concept (formula) runs* for φ as sequences $r = \dots r(-1)\, r(0)\, r(1) \dots$ such that each subsequence of length $2\ell_\varphi + 1$ is a concept (formula) run segment, and, for all $n \in \mathbb{Z}$,

R5 for all $\alpha\,\mathcal{U}\beta \in \mathsf{cl}^*(\varphi)$, we have $\alpha\,\mathcal{U}\beta \in r(n)$ iff r realises $\alpha\,\mathcal{U}\beta$ at position n, and
R6 for all $\alpha\,\mathcal{S}\beta \in \mathsf{cl}^*(\varphi)$, we have $\alpha\,\mathcal{S}\beta \in r(n)$ iff r realises $\alpha\,\mathcal{S}\beta$ at position n.

These conditions take care of satisfying temporal concepts and formulae over an infinite time horizon.

We still need to check whether a set of concept runs (or run segments) can actually be composed into a coherent model. In particular, we have to be aware of (interval-rigid) role connections

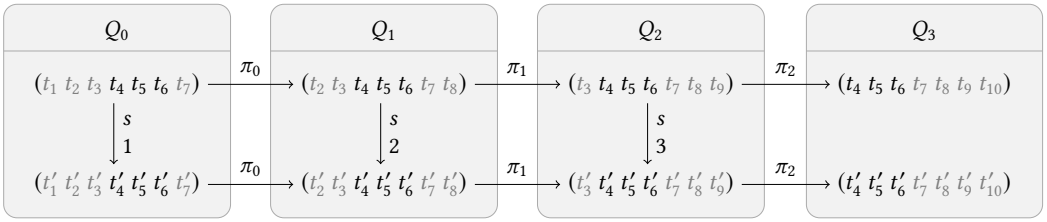


Fig. 1. Illustration of role constraints and compatibility relations.

between elements. In our abstraction, these connections are represented using *role constraints*, which connect two run segments with a role, and a counter used to keep track of how long domain elements corresponding to those run segments have been connected. Formally, a role constraint for φ is an expression of the form $\sigma \overset{s}{k} \sigma'$, where σ, σ' are concept run segments, $s \in \mathsf{N_R}(\varphi)$, and $k \in [1, \mathsf{iRig}(s)]$ such that

C1 $\{\neg C \mid \neg\exists s.C \in \sigma(0)\} \subseteq \sigma'(0)$.

Intuitively, $\sigma \overset{s}{k} \sigma'$ states that the domain elements described by $\sigma(0)$ and $\sigma'(0)$ are connected by the role s at the current time point, have been connected by s in the previous $k-1$ time points, and will remain connected in the next $\mathsf{iRig}(s) - k$ time points. If two domain elements are connected by s for more than $\mathsf{iRig}(s)$ time points, then there might be more than one role constraint in the same quasistate expressing this connection. Condition **C1** ensures that, if $\sigma(0)$ cannot have any s-successors that satisfy C, then $\sigma'(0)$ does not satisfy C.

Quasistates describe the behaviour of a whole interpretation and its elements at a single time point, and incorporate bounded information about up to ℓ_φ time points of the past and future. Formally, a *quasistate* for φ is a pair $Q = (\mathcal{R}_Q, \mathcal{C}_Q)$, where $\mathcal{R}_Q$ is a set of run segments and $\mathcal{C}_Q$ a set of role constraints over $\mathcal{R}_Q$ such that

S1 $\mathcal{R}_Q$ contains exactly one formula run segment, denoted by σ_Q in the following,
S2 for all $\sigma \in \mathcal{R}_Q$ and $\exists s.D \in \sigma(0)$, there is $\sigma \overset{s}{k} \sigma' \in \mathcal{C}_Q$ with $D \in \sigma'(0)$ and $k \in [1, \mathsf{iRig}(s)]$, and
S3 for all $C \sqsubseteq D \in \mathsf{cl}^{\mathsf{f}}(\varphi)$, we have $C \sqsubseteq D \in \sigma_Q(0)$ iff $C \in \sigma(0)$ implies $D \in \sigma(0)$ for all concept run segments $\sigma \in \mathcal{R}_Q$.

In order to integrate quasistates into an infinite quasimodel, we need conditions stating when this is possible. To this end, we consider a pair (Q, Q') of quasistates to be *compatible* if there is a *compatibility relation* $\pi \subseteq \mathcal{R}_Q \times \mathcal{R}_{Q'}$ such that

C2 every run segment in $\mathcal{R}_Q$ and $\mathcal{R}_{Q'}$ occurs in the domain and range of π, respectively,
C3 each pair $(\sigma, \sigma') \in \pi$ satisfies $\sigma^{>-\ell_\varphi} = \sigma'^{<\ell_\varphi}$,
C4 for all $(\sigma_1, \sigma'_1) \in \pi$ and $\sigma_1 \overset{s}{k} \sigma_2 \in Q$ with $k < \mathsf{iRig}(s)$, there is $\sigma'_1 \overset{s}{k+1} \sigma'_2 \in Q'$ with $(\sigma_2, \sigma'_2) \in \pi$, and
C5 for all $(\sigma_1, \sigma'_1) \in \pi$ and $\sigma'_1 \overset{s}{k} \sigma'_2 \in Q'$ with $k > 1$, there is $\sigma_1 \overset{s}{k-1} \sigma_2 \in Q$ with $(\sigma_2, \sigma'_2) \in \pi$.

These relations ensure that we can combine run segments of consecutive quasistates such that the interval-rigid roles are respected. Observe that **C3** also makes sure that the unique formula run segments match each other. Moreover, the set of all compatibility relations for a pair of quasistates (Q, Q') is closed under union, which means that compatible quasistates always have a unique maximal compatibility relation (w.r.t. set inclusion).

To illustrate this, consider Figure 1, which shows a finite sequence of pairwise compatible quasistates, each containing two run segments. Here, $\ell_\varphi = \mathsf{iRig}(s) = 3$. The relations π_0, π_1, and π_2

satisfy Conditions **C2**–**C5**, which, together with **C1**, ensure that a run going through the types t_4, t_5, and t_6 can be connected to another run via the role s for at least 3 consecutive time points.

A quasistate is the abstraction of an interpretation at a single time point, and compatibility describes when two quasistates can capture consecutive interpretations. Consequently, a *quasimodel* for φ is a pair $(S, \mathfrak{R})$, where S is an infinite sequence of quasistates $\dots S(-1)\, S(0)\, S(1) \dots$ with consecutive quasistates being compatible and $\mathfrak{R}$ is a set of runs such that

M1 $\varphi \in \sigma_{S(0)}(0)$,
M2 the runs in $\mathfrak{R}$ are of the form $\dots \sigma_{-1}(0)\, \sigma_0(0)\, \sigma_1(0)\, \sigma_2(0) \dots$ such that, for every $i \in \mathbb{Z}$, we have $(\sigma_i, \sigma_{i+1}) \in \pi_i$, where π_i is the maximal compatibility relation for the pair $(S(i), S(i+1))$, and
M3 for every $\sigma \in \mathcal{R}_{S(i)}$, there exists a run $r \in \mathfrak{R}$ with $r^{[i-\ell_\varphi, i+\ell_\varphi]} = \sigma$.

Condition **M2** ensures that the runs $\dots \sigma_{-1}(0)\, \sigma_0(0)\, \sigma_1(0)\, \sigma_2(0) \dots$ contain the run segments $\dots, \sigma_{-1}$, $\sigma_0, \sigma_1, \sigma_2, \dots$: because σ_{-1}, σ_0, σ_1, and σ_2 stand in a compatibility relation with each other, we have $\sigma_{-1}(0) = \sigma_0(-1)$, $\sigma_1(0) = \sigma_0(1)$, $\sigma_2(0) = \sigma_0(2)$, and so on. Moreover, together with **M3**, Condition **M2** guarantees that $\mathfrak{R}$ always contains exactly one formula run, and that every run segment has a corresponding run. Condition **M1** ensures that φ, the formula we are checking, is present at position 0 of the unique formula run in $\mathfrak{R}$.

We can show that every quasimodel describes a satisfying interpretation for φ and, conversely, that every such interpretation can be abstracted to a quasimodel. Moreover, one can always find a quasimodel of a regular shape. This is captured by the following lemma, proven in Appendix B.

Lemma 4.3. *An* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formula* φ *is satisfiable with interval-rigid names iff* φ *has a quasimodel* $(S, \mathfrak{R})$ *in which* S *is of the form*

$$^{\omega}\big(S(-k_1) \dots S(-k_2 - 1)\big)\, S(-k_2) \dots S(0) \dots S(k_3)\, \big(S(k_3 + 1) \dots S(k_4)\big)^{\omega},$$

where k_1, k_2, k_3 *and* k_4 *are bounded triple exponentially in the size of* φ *and* $\mathsf{iRig}|_\varphi$.

Recall that we assume all numbers in the function $\mathsf{iRig}|_\varphi$ to be given in binary notation. Lemma 4.3 allows us to devise a 2-ExpSpace decision procedure for satisfiability of a given $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formula, similar to the setting without interval-rigid names from Section 3, where we guess the indices k_1, k_2, k_3, k_4, as well as all quasistates, one after the other. Since 2-ExpSpace and 2-NExpSpace coincide, we can describe the procedure in a non-deterministic manner. The guessed numbers, k_1, k_2, k_3, and k_4, each take at most double exponential space in binary encoding. To verify that the guessed sequence $S(-k_1) \dots S(k_4)$ of quasistates corresponds to a quasimodel as in Lemma 4.3, only three quasistates have to be kept in memory at any time. Observe that the size of each quasistate is double exponentially bounded in the size of the input. Additionally, for each run segment $\sigma \in \mathcal{R}_{S(k_3)}$, we keep in memory a list of the $\mathcal{U}$-expressions in $\sigma(0)$, which need to be satisfied in the repeating part of the quasimodel (for details, see Appendix B). The size of each list is polynomial in the size of the input, and since there are at most double exponentially many run segments in $\mathcal{R}_{S(k_3)}$, this also requires (at most) double exponential space. We do the same with $\mathcal{R}_{S(-k_2)}$ and $\mathcal{S}$-expressions.

A matching hardness result without interval-rigid names was shown by Artale et al. [12].

Theorem 4.4. *Satisfiability in* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *with interval-rigid names is 2-ExpSpace-complete.*

4.2 Satisfiability with Global GCIs is 2-ExpTime-complete

We now consider $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$, that is, we disallow the use of temporal operators on GCIs. Temporal operators on GCIs are very powerful because they allow universal constraints to change over time.

For example,

$$\Box_{(\infty,-365)}\big(\exists \texttt{finding.Pneumonia} \sqsubseteq \exists \texttt{recommended.}(\texttt{Aminopenicillin} \sqcup \texttt{Fluoroquinolone})\big) \wedge$$
$$\Box_{[-365,0]}\big(\exists \texttt{finding.Pneumonia} \sqsubseteq \exists \texttt{recommended.Aminopenicillin}\big)$$

expresses that fluoroquinolones were removed from the list of recommended antibiotics for pneumonia treatment one year ago. If we allow only global GCIs, we cannot express such changing recommendations; however, as we show in this section, the complexity drops from 2-ExpSpace to 2-ExpTime. Here, we can still use temporal operators on assertions, which is central if we want to reason about temporal data, as well as on concepts. Note that Example 4.1 uses only temporal operators on assertions.

For satisfiability of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$ formulae, we show 2-ExpTime-completeness. By Theorem 3.1, we can w.l.o.g. consider $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formulae of the form $\varphi = \Box\Box^{-}\mathcal{T}$, where $\mathcal{T}$ is a conjunction of GCIs. Due to the definition of quasimodels, this means that all formula types in a quasimodel must contain all GCIs from $\mathcal{T}$. To prove our upper bound (Theorem 4.6), we show that, to check satisfiability of φ with interval-rigid roles, it suffices to consider *constant* quasimodels $(S, \mathfrak{R})$ with S of the form ${}^{\omega}Q^{\omega}$, where Q is a quasistate (cf. Lemma 3.6).

Lemma 4.5. *An* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$ *formula* φ *is satisfiable over* $\mathbb{Z}$ *with interval-rigid names iff there is a constant quasimodel for* φ*.*

Proof. By Lemma 4.3, it suffices to look for a quasimodel for φ. Given such a quasimodel $(S, \mathfrak{R})$, we can modify all $S(n)$ by adding to $S(n)$ all concept run segments and role constraints in $S(i)$, with $i \neq n$. Denote the resulting sequence by S'. Since each $S(i)$ satisfies **S2**, each modified $S'(n)$ satisfies **S2**. As all GCIs are global, **S3** holds. Since we do not change formula run segments, Conditions **S1** and **M1** for the existence of a quasimodel still hold. Thus, S' is a sequence of quasistates. Moreover, each $S'(n)$ is compatible with $S'(n+1) = S'(n)$. To satisfy Conditions **M2** and **M3**, we add to $\mathfrak{R}$ all runs in

$$\{r^{\rightarrow i} \mid r \in \mathfrak{R} \text{ a concept run}, i \in \mathbb{Z}\},$$

where $r^{\rightarrow i}(n) := r(n+i)$ for all $r \in \mathfrak{R}$ and $n \in \mathbb{Z}$, resulting in the set $\mathfrak{R}'$ of runs. The tuple $(S', \mathfrak{R}')$ is still a quasimodel and satisfies $S'(i) = S'(j)$ for all $i, j \in \mathbb{Z}$, as required. □

To solve the satisfiability problem, we describe a type elimination procedure for checking the existence of a constant quasimodel. We first compute in 2-ExpTime the set $\mathrm{rs}(\varphi)$ of concept run segments σ for φ such that $C \in \sigma(0)$ implies $D \in \sigma(0)$, for all $C \sqsubseteq D \in \mathrm{cl}^{\mathsf{f}}(\varphi)$. We then compute the set $\mathrm{rc}(\varphi)$ of all possible role constraints over $\mathrm{rs}(\varphi)$. We initialise $Q = (\mathcal{R}_Q, \mathcal{C}_Q)$ with $\mathcal{R}_Q = \mathrm{rs}(\varphi)$ and $\mathcal{C}_Q = \mathrm{rc}(\varphi)$.

We then proceed as follows. Given a concept run segment $\sigma \in \mathcal{R}_Q$, let X_σ be the set of all images of functions mapping each concept of the form $\exists s.D \in \sigma(0)$ to a role constraint $\sigma \overset{s}{k} \sigma' \in \mathcal{C}_Q$ with $D \in \sigma'(0)$. Given $x_\sigma \in X_\sigma$ and $x_{\sigma'} \in X_{\sigma'}$, we say that $(x_\sigma, x_{\sigma'})$ is *suitable* if $\sigma^{>-\ell_\varphi} = \sigma'^{<\ell_\varphi}$ and

- $\sigma \overset{s}{k} \sigma_1 \in x_\sigma$ with $k < \mathrm{iRig}(s)$ implies that there is $\sigma' \overset{s}{k+1} \sigma_2 \in x_{\sigma'}$ with $\sigma_1^{>-\ell_\varphi} = \sigma_2^{<\ell_\varphi}$; and
- $\sigma' \overset{s}{k} \sigma_2 \in x_{\sigma'}$ with $k > 1$ implies that there is $\sigma \overset{s}{k-1} \sigma_1 \in x_\sigma$ with $\sigma_1^{>-\ell_\varphi} = \sigma_2^{<\ell_\varphi}$.

We define a relation $\pi \subseteq \mathcal{R}_Q \times \mathcal{R}_Q$, where π is a set of pairs (σ, σ') such that, for some $x_\sigma \in X_\sigma$ and $x_{\sigma'} \in X_{\sigma'}$, the pair $(x_\sigma, x_{\sigma'})$ is suitable. We then exhaustively eliminate each run segment σ from $\mathcal{R}_Q$ that violates one of the following conditions:

E1 there are $\sigma_1, \sigma_2 \in \mathcal{R}_Q$ such that $(\sigma_1, \sigma) \in \pi$ and $(\sigma, \sigma_2) \in \pi$,

E2 for all $C\,\mathcal{U}_I D \in \sigma(0)$, there is $k > 1$ and a sequence $\sigma_2, \dots, \sigma_k \in \mathcal{R}_Q$ such that $\sigma(0)\,\sigma_2(0) \cdots \sigma_k(0)$ realises $C\,\mathcal{U}_I D$, $(\sigma, \sigma_2) \in \pi$, and $(\sigma_\ell, \sigma_{\ell+1}) \in \pi$, for all $1 < \ell < k$, and

E3 for all $C\,\mathcal{S}_I D \in \sigma(0)$, there is $k > 1$ and a sequence $\sigma_k, \dots, \sigma_2 \in \mathcal{R}_Q$ such that $\sigma_k(0) \cdots \sigma_2(0)\,\sigma(0)$ realises $C\,\mathcal{S}_I D$, $(\sigma_2, \sigma) \in \pi$, and $(\sigma_{\ell+1}, \sigma_\ell) \in \pi$, for all $1 < \ell < k$.

We assume that, whenever $\sigma \in \mathcal{R}_Q$ is eliminated, we remove all role constraints involving σ from $\mathcal{C}_Q$ and, for all remaining $\sigma' \in \mathcal{R}_Q$, we update $X_{\sigma'}$ accordingly. We also remove (σ_1, σ_2) from π if there is no $x_{\sigma_1} \in X_{\sigma_1}$ and $x_{\sigma_2} \in X_{\sigma_2}$ such that $(x_{\sigma_1}, x_{\sigma_2})$ is suitable.

When this process terminates, we have found the maximal sets of concept run segments and role constraints to form the quasistate of a constant quasimodel (if a quasimodel exists). Our algorithm returns "satisfiable" iff there is a surviving concept run segment in $\mathcal{R}_Q$. Theorem 4.6 formalises the upper bound obtained using this algorithm.

Theorem 4.6. *Satisfiability in* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$ *with interval-rigid names is in 2-ExpTime.*

Proof. Since $\mathcal{R}_Q$ and π contain at most double exponentially many elements, Condition **E1** can be checked in double exponential time. For Conditions **E2** and **E3**, the algorithm performs a series of reachability checks in the graph $(\mathcal{R}_Q, \pi)$, which is of double exponential size. We explain in more detail how to deal with $\mathcal{U}$-expressions. The procedure for $\mathcal{S}$-expressions is analogous. For each $\sigma \in \mathcal{R}_Q$, we consider the set U that contains all (polynomially many) until expressions in $\sigma(0)$. We enumerate all possible total orders $C_1\,\mathcal{U}_{I_1} D_1 < C_2\,\mathcal{U}_{I_2} D_2 < \cdots < C_{|U|}\,\mathcal{U}_{I_{|U|}} D_{|U|}$ over U, of which there are exponentially many. For a fixed such order, we enumerate all possible choices of run segments $\sigma^1, \dots, \sigma^{|U|} \in \mathcal{R}_Q$ such that each σ^{i+1} is reachable from σ^i, for all $i \in [1, |U|)$, and the $\mathcal{U}$-expression $C_i\,\mathcal{U}_{I_i} D_i$ is satisfied on the path to σ^i, for all $i \in [1, |U|]$. As before, there are double exponentially many possibilities for these run segments (since $|U|$ is polynomial), and the reachability checks can also be done in double exponential time. As there are initially double exponentially many run segments and in every step some run segment is eliminated, the procedure terminates in double exponential time.

Correctness of the procedure is given by the following claim.

Claim. *The algorithm returns "satisfiable" iff φ is satisfiable with interval-rigid names.*

By Lemma 4.5, it suffices to check the existence of a constant quasimodel. ($\Rightarrow$) It is straightforward to verify that any sequence resulting from a successful series of the checks described above satisfies the conditions of Lemma 4.5. In particular, any two consecutive quasistates are compatible by Condition **E1** and the definition of π. Moreover, the required runs exist by Conditions **E2** and **E3**. ($\Leftarrow$) Let S be a sequence as in Lemma 4.5. Then the concept run segments and role constraints of all $S(i)$ must be included in Q as constructed by the algorithm, because the conditions checked by the algorithm are satisfied by any such constant quasimodel. □

The matching lower bound can be shown even without metric temporal operators by an easy adaptation of an existing proof [20], which uses a rigid role to fix an interpretation structure over which information can be transferred. However, this structure only needs to stay fixed for exponentially many time points, which can be ensured using an interval-rigid role instead.

Theorem 4.7. *Satisfiability in* $\mathrm{LTL}_{\mathcal{ALC}|gGCI}$ *with interval-rigid names is 2-ExpTime-hard.*

5 $\mathcal{ALC}$-LTL$^{\mathsf{bin}}$ WITH INTERVAL-RIGID NAMES

After considering the very expressive DL $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$, we now focus on its sublogic $\mathcal{ALC}$-LTL$^{\mathsf{bin}}$, the fragment of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ that does not allow temporal concept operators. The first study on the even smaller logic $\mathcal{ALC}$-LTL [20] provided tight complexity bounds for all possible combinations of rigid concept and role names over the natural numbers. In contrast to $\mathrm{LTL}_{\mathcal{ALC}}$, rigid concepts cannot be simulated by GCIs, and rigid roles do not immediately lead to undecidability. In this section,

we consider interval-rigid names in addition to rigid names, and show several new complexity lower bounds that already apply to $\mathcal{ALC}$-LTL, i.e., without metric temporal operators. We prove our lower bounds for $\mathcal{ALC}$-LTL with past operators over the integers, but similar arguments also apply for the case where we have natural numbers and no past operators [16]. Following the first work on $\mathcal{ALC}$-LTL [20], we denote restriction of $\mathcal{ALC}$-LTL$^{\text{bin}}$ to global GCIs by $\mathcal{ALC}$-LTL$^{\text{bin}}_{|gGCI}$.

The insight that interval-rigid concepts can express the operator $\bigcirc$ on the concept level is central for our hardness proofs. This directly yields our first result, namely ExpSpace-hardness for $\mathcal{ALC}$-LTL with interval-rigid concepts, in Section 5.1. This observation is also central to Section 5.2, where we show 2-ExpSpace-hardness for $\mathcal{ALC}$-LTL with interval-rigid roles, matching the upper bound provided by Theorem 4.4. We leave open the precise complexity if only global GCIs are allowed in that setting. Finally, if interval-rigid concepts and rigid roles are allowed, we are able to show undecidability (see Section 5.3).

5.1 Rigid and Interval-Rigid Concepts

As the first setting, we consider the case where only concept names can be rigid or interval-rigid (but not role names). Recall from Section 4 that rigid concepts and interval-rigid concepts are expressible in $\text{LTL}^{0,\infty}_{\mathcal{ALC}}$ via global GCIs, and hence by Theorems 3.4 and 3.7 we can transfer all upper bounds from $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ without (interval-)rigid roles. Moreover, the ExpTime-hardness from $\mathcal{ALC}$ [43] and the ExpSpace-hardness from LTL$^{\text{bin}}$ [3] still apply, which already gives us tight bounds for satisfiability in $\mathcal{ALC}$-LTL$_{|gGCI}$ and $\mathcal{ALC}$-LTL$^{\text{bin}}_{|gGCI}$. This leaves only two gaps to fill. First, for $\mathcal{ALC}$-LTL with (interval-)rigid concepts, the best known lower bound from the literature is NExpTime [20] (for the natural numbers and without past operators), which we improve here to a tight ExpSpace lower bound (see Theorem 5.2). Second, we show that, in contrast to $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$, this complexity does not increase when we add metric temporal operators. That is, satisfiability in $\mathcal{ALC}$-LTL$^{\text{bin}}$ with (interval-)rigid concepts is also ExpSpace-complete (see Theorem 5.3).

As a first step, we describe how to simulate the operator $\bigcirc$ on concepts by using interval-rigid concept names. For this purpose, we consider $\mathcal{ALC}$-LTL$^{\bigcirc}$ formulae, which extend $\mathcal{ALC}$-LTL by allowing the concept constructor $\bigcirc C$. However, since we do not need past operators for our hardness proofs, in the following we consider $\mathcal{ALC}$-LTL$^{\bigcirc}$ formulae only without past operators and w.r.t. a temporal semantics based on $\mathbb{N}$.

Lemma 5.1. *Any $\mathcal{ALC}$-LTL$^{\bigcirc}$ formula φ can be translated in polynomial time into an $\mathcal{ALC}$-LTL formula ψ' with interval-rigid concept names such that φ is satisfiable over $\mathbb{N}$ iff φ' is satisfiable over $\mathbb{Z}$.*

Proof. The basic idea is to use 2-rigid concept names $A_{\bigcirc D}$ to transfer the information that a concept $\bigcirc D$ should be satisfied at time point i to the next time point $i+1$, at which we then enforce that D is satisfied. However, due to the definition of interval-rigid concept names, we have to ensure that $A_{\bigcirc D}$ was not already satisfied at $i-1$, because then it does not need to be satisfied at $i+1$. To solve this problem, we use *three* concept names $A^j_{\bigcirc D}$, $0 \leq j \leq 2$. Each of them can be used only to transfer information from any time point i that is congruent to j modulo 3 to the next time point $i+1$. At time point $i+2$, we then enforce that $A^j_{\bigcirc D}$ is unsatisfiable, which means that we can reuse it at time point $i+3$, which is again congruent to j modulo 3. For the transitions from $i+1$ to $i+2$ and from $i+2$ to $i+3$, we can use the other two concept names (see Figure 2).

To start, we set up a counter modulo 3, by marking all domain elements of a model of our formula with the concept name $A^{(i \bmod 3)}$. For convenience, we consider all superscripts modulo 3, i.e., we have $A^3 = A^0$ and $A^{-1} = A^2$:

$$\psi_{\mathsf{counter}} := \big(\top \sqsubseteq A^0\big) \wedge \Box\Box^- \bigwedge_{0 \leq i \leq 2} \Big(\big(\big(\top \sqsubseteq A^i\big) \rightarrow \bigcirc\big(\top \sqsubseteq A^{i+1}\big)\big) \wedge \big(A^i \sqsubseteq \neg A^{i+1}\big)\Big).$$

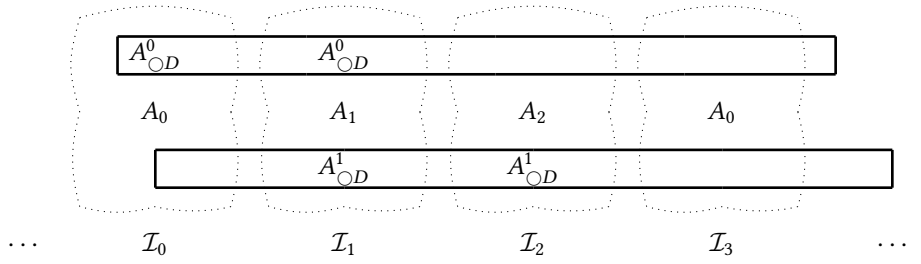


Fig. 2. A model with an element in $(A^0_{\bigcirc D})^{\mathcal{I}_0}$ (upper rectangle) and an element in $(A^1_{\bigcirc D})^{\mathcal{I}_1}$ (lower rectangle).

This formula implies that all domain elements satisfy exactly one of the concept names A^0, A^1, A^2: at time point 0, all of them satisfy A^0, afterwards A^1, and so on (see Figure 2).

We now define the operator $\cdot^{-\bigcirc}$ that transforms any $\mathcal{ALC}$-LTL$^\bigcirc$ formula φ or concept C into a formula or concept of $\mathcal{ALC}$-LTL. Consider first the case of an $\mathcal{ALC}$-LTL$^\bigcirc$ concept C, i.e., an LTL$_{\mathcal{ALC}}$ concept that contains no temporal operators except $\bigcirc$. We replace all its subconcepts of the form $\bigcirc D$, starting from the *outermost* ones, i.e., the ones that do not occur in the scope of another $\bigcirc$-operator. To obtain $C^{-\bigcirc}$, we replace each outermost subconcept $\bigcirc D$ by

$$\bigsqcup_{0 \le i \le 2} (A^i_{\bigcirc D} \sqcap A^i),$$

where $A^i_{\bigcirc D}$, $0 \le i \le 2$, are fresh 2-rigid concept names. The resulting concept $C^{-\bigcirc}$ is a classical $\mathcal{ALC}$ concept. As described above, $A^i_{\bigcirc D}$ indicates that D should hold at the next time point, but it can only be used at time points marked with A^i. We can thus simulate the semantics of $\bigcirc D$ as follows:

$$\psi_{\bigcirc D} := \Box\Box^- \bigwedge_{0 \le i \le 2} \left(\left(A^{i-1} \sqsubseteq \neg A^i_{\bigcirc D}\right) \wedge \left(A^{i+1} \sqsubseteq A^i_{\bigcirc D} \leftrightarrow D^{-\bigcirc}\right) \right).$$

Note that the replacement operator $\cdot^{-\bigcirc}$ is applied here to the inner concept D, which may still contain more occurrences of $\bigcirc$. As observed above, $D^{-\bigcirc}$, and therefore also $\psi_{\bigcirc D}$, can be formulated in pure $\mathcal{ALC}$-LTL.

The formula $\psi_{\bigcirc D}$ says that $A^i_{\bigcirc D}$ is unsatisfiable at time points marked by A^{i-1}. Since $A^i_{\bigcirc D}$ is 2-rigid, if $A^i_{\bigcirc D} \sqcap A^i$ is satisfied by some individual, then at the next time point (marked by A^{i+1}) this individual must still satisfy $A^i_{\bigcirc D}$, and hence also $D^{-\bigcirc}$. Conversely, if an individual satisfies $D^{-\bigcirc}$ and A^{i+1}, then it must also satisfy $A^i_{\bigcirc D}$, and therefore it must have satisfied $A^i_{\bigcirc D}$ at the previous time point. This means that satisfying $A^i_{\bigcirc D}$ at any time point marked by A^i is equivalent to satisfying $D^{-\bigcirc}$ at the next time point. It is easy to show by induction on the structure of concepts that, under the constraints given by ψ and all formulae $\psi_{\bigcirc D}$ for which $\bigcirc D \in \mathrm{sub}^{\mathrm{c}}(C)$, the concept $C^{-\bigcirc}$ is thus equivalent to C.

Given any $\mathcal{ALC}$-LTL$^\bigcirc$ *formula* φ, we obtain $\varphi^{-\bigcirc}$ by replacing all outermost concepts C by $C^{-\bigcirc}$. It is easy to see that φ is satisfiable iff the $\mathcal{ALC}$-LTL formula

$$\varphi^{-\bigcirc} \wedge \psi_{\mathsf{counter}} \wedge \bigwedge_{\bigcirc D \in \mathrm{sub}^{\mathrm{c}}(\varphi)} \psi_{\bigcirc D}$$

is satisfiable. Since satisfiability of φ is considered w.r.t. $\mathbb{N}$, and we need the past operator $\Box^-$ here only to ensure that $A^0_{\bigcirc D}$ cannot be satisfied at time point -1, the time points before -1 can be ignored. □

We now obtain our first lower bound via a result from [30], which shows that satisfiability in $\mathcal{ALC}$-LTL$^{\bigcirc}$ is ExpSpace-hard.

Theorem 5.2. *Satisfiability in $\mathcal{ALC}$-LTL with interval-rigid concepts is ExpSpace-hard.*

Proof. The temporal logic $\mathcal{QTL}^1$ is the 1-variable fragment of the extension of first-order logic with the *strict* version of the $\mathcal{U}$-operator [30], which can equivalently be expressed by the non-strict $\mathcal{U}$ (as we use it here) and $\bigcirc$. Furthermore, $\mathcal{QTL}^1_{\textcircled{1}}$ is the fragment of $\mathcal{QTL}^1$ in which only $\bigcirc$ can be applied to open formulae, while $\mathcal{U}$ can be applied to sentences only. It is known that satisfiability in $\mathcal{QTL}^1_{\textcircled{1}}$ is ExpSpace-complete, and moreover the proof of that result uses only unary predicates [30, Theorem 11.33]. Monadic $\mathcal{QTL}^1_{\textcircled{1}}$ formulae can be expressed as $\mathcal{ALC}$-LTL$^{\bigcirc}$ formulae in the obvious way, where open formulae correspond to concepts and sentences correspond to formulae. For example, a formula of the form $(\forall x.\varphi)$ can be expressed as the GCI $\top \sqsubseteq C_\varphi$, where C_φ is the concept equivalent to the open formula φ. The translation of the other constructors is straightforward. Together with Lemma 5.1, this shows the claim. □

To show the matching upper bound for $\mathcal{ALC}$-LTL$^{\text{bin}}$, we employ a simple reduction to use the results shown for LTL$_{\mathcal{ALC}}$ [48] (see also Theorem 3.7). The idea is that we first eliminate the metric temporal operators from the $\mathcal{ALC}$-LTL$^{\text{bin}}$ formula by using Proposition 3.3, which yields an $\mathcal{ALC}$-LTL formula with an exponential number of subformulae. Moreover, as described in Section 4, we can express (interval-)rigid concept names by using LTL$^{0,\infty}_{\mathcal{ALC}}$ concepts. Using Theorem 3.4, we can obtain an LTL$_{\mathcal{ALC}}$ formula with only an additional polynomial blow-up. It then suffices to observe that the complexity of this approach is not affected by the exponential blow-up in the number of subformulae.

Theorem 5.3. *Satisfiability in $\mathcal{ALC}$-LTL$^{\text{bin}}$ with rigid concepts and interval-rigid concepts is in ExpSpace.*

Proof. Let φ be an $\mathcal{ALC}$-LTL$^{\text{bin}}$ formula, and $\varphi^\dagger$ the $\mathcal{ALC}$-LTL formula obtained from φ by the construction of Proposition 3.3. Since φ does not contain any temporal *concepts*, the number of subformulae of φ increases at most exponentially, but the subconcepts of φ stay the same (although they can occur more often). More precisely, we have $|\mathsf{sub}^{\mathsf{f}}(\varphi^\dagger)| \leq (\ell_\varphi + 1)^3 |\mathsf{sub}^{\mathsf{f}}(\varphi)|$ and $\mathsf{sub}^{\mathsf{c}}(\varphi^\dagger) = \mathsf{sub}^{\mathsf{c}}(\varphi)$, where ℓ_φ again denotes the maximal number occurring in temporal operators or interval-rigidity constraints. For example, the translation of $\rho\,\mathcal{U}_{[c_1,c_2]}\psi$ uses formulae of the form $\bigcirc^i\psi$ (with $i \leq \ell_\varphi$), which have at most $\ell_\varphi + 1$ subformulae $\bigcirc^i\psi, \bigcirc^{i-1}\psi, \ldots, \bigcirc\psi, \psi$, where before we had only ψ itself. These formulae are further used in a disjunction of conjunctions (each with at most $\ell_\varphi + 1$ components), hence the factor of $(\ell_\varphi + 1)^3$ above. Note that the subformulae of ρ and ψ remain the same during this replacement step. If we do this in a "bottom-up" way, starting with the innermost subformulae, in each step we replace one subformula by $(\ell_\varphi + 1)^3$ new ones, and can do this at most once for each $\mathcal{U}$-formula, i.e., at most $|\mathsf{sub}^{\mathsf{f}}(\varphi)|$ times.

We extend $\varphi^\dagger$ by global GCIs that simulate the (interval-)rigidity constraints using LTL$^{0,\infty}_{\mathcal{ALC}}$ concepts, as described in Section 4, and denote the result by $\varphi^\ddagger$. We have $|\mathsf{sub}^{\mathsf{f}}(\varphi^\ddagger)| \leq p(|\mathsf{sub}^{\mathsf{f}}(\varphi^\dagger)| + m)$ and $|\mathsf{sub}^{\mathsf{c}}(\varphi^\ddagger)| \leq p(|\mathsf{sub}^{\mathsf{c}}(\varphi^\dagger)| + m)$, where $m \leq |\mathsf{sub}^{\mathsf{c}}(\varphi)|$ is the number of (interval-)rigid concept names occurring in φ and p is a polynomial. We can now assume that $\mathsf{N}_{\mathsf{Rig}}$ and $\mathsf{N}_{\mathsf{IRig}}$ are empty. By Theorem 3.4, we can express $\varphi^\ddagger$ as an LTL$_{\mathcal{ALC}}$ formula φ' with $|\mathsf{sub}^{\mathsf{c}}(\varphi')| \leq p'(|\mathsf{sub}^{\mathsf{c}}(\varphi^\ddagger)|)$ and $|\mathsf{sub}^{\mathsf{f}}(\varphi')| \leq p'(|\mathsf{sub}^{\mathsf{f}}(\varphi^\ddagger)|)$ for some polynomial p'. In summary, we have obtained a formula φ' that is equisatisfiable to φ, and to which we can apply Theorem 3.7 (where we state that the well-known quasimodel algorithm [48] can be adapted to $\mathbb{Z}$) in order to decide satisfiability. Moreover, the number of subformulae of φ' is bounded exponentially in the size of the original formula, while the number of subconcepts increased only by a polynomial. It thus suffices to find a regular quasimodel

for which the length of the initial and repeating parts is bounded by a function that is double exponential in the size of $\mathsf{sub}^{\mathsf{c}}(\varphi')$, exponential in the size of $\mathsf{sub}^{\mathsf{f}}(\varphi')$, and polynomial in the size of $\mathsf{N}_{\mathsf{I}}(\varphi') = \mathsf{N}_{\mathsf{I}}(\varphi)$, which is overall at most double exponential in the size of the original formula φ [48, Theorem 24] [30, Theorem 11.30]. Since the size of each quasistate is bounded exponentially in the size of φ [48, Definition 10], we only need exponential space to search for such a quasimodel. □

5.2 Interval-Rigid Roles

We now additionally allow interval-rigid roles and obtain a 2-ExpSpace-hardness result matching the upper bound given by Theorem 4.4. By Lemma 5.1, it suffices to prove hardness for $\mathcal{ALC}$-LTL$^{\bigcirc}$ over $\mathbb{N}$. The proof uses a reduction from the word problem for double exponentially space bounded deterministic Turing machines. We encode the computation of the machine along the temporal dimension using double exponentially many time points for each configuration. The main problem is that we need to transfer information about the tape cells double exponentially far to the next configuration. We solve this by encoding a double exponential counter with exponentially many bits, where each time point corresponds to a bit position, and the counter is incremented using names that are interval-rigid for exponentially many time points. To transfer information over double exponentially many steps, we use a domain element for each tape address and check whether there is a match between the current value of the double exponential counter and the address represented by this domain element. Lower bounds using double exponential counters have already been studied [33, Theorem 4.2]. However our proof using interval-rigid roles uses different constructions, presented in full detail in Appendix C.

Theorem 5.4. *Satisfiability in $\mathcal{ALC}$-LTL with interval-rigid names is 2-ExpSpace-hard.*

5.3 Rigid Roles and Interval-Rigid Concepts

Finally, we show that further adding rigid roles makes satisfiability in $\mathcal{ALC}$-LTL undecidable, even if temporal operators are only used on assertions and no interval-rigid roles are used. Our proof is by a reduction from the following tiling problem, which is Σ^1_1-hard [34].[4]

> Given a finite set of tile types T with horizontal and vertical compatibility relations H and V, respectively, and $t_0 \in T$, decide whether one can tile $\mathbb{Z} \times \mathbb{N}$ with t_0 appearing infinitely often in the first row to the right of position $(0, 0)$.

We define an $\mathcal{ALC}$-LTL$_{|gGCI}$ formula φ_{T,t_0} that expresses this property. The vertical dimension ($\mathbb{N}$) is expressed using an infinite rigid role chain. For the horizontal dimension ($\mathbb{Z}$), we use the temporal dimension and simulate the constructor $\bigcirc$ by using ideas similar as in Lemma 5.1. Since we can use only global GCIs, however, we cannot express the progression of the counter (using A^0, A^1, A^2) over *all* domain elements simultaneously. However, it is sufficient to do this for all elements on the rigid role chain mentioned above, which can be done by transferring the value of the counter using value restrictions.

More precisely, our reduction uses the following symbols:

- a rigid role name r to encode the vertical dimension of the $\mathbb{Z} \times \mathbb{N}$ grid;
- flexible concept names A^0, A^1, A^2 to encode a counter modulo 3 along the horizontal (temporal) dimension;
- flexible concept names P_t (with $t \in T$) to denote the current tile type;
- 2-rigid concept names N^0_t, N^1_t, N^2_t for the horizontally adjacent tile type; and

[4] In [34], the problem is shown to be Σ^1_1-hard for the quadrant $\mathbb{N} \times \mathbb{N}$, but this result can be extended to $\mathbb{Z} \times \mathbb{N}$ exactly as in the proof of Theorem 6.3 in the same paper. Essentially, the runs of Turing machines are extended to trivial infinite "backward" computations, which do not otherwise affect the behaviour of the machines.

- an individual name a that denotes the first row of the grid.

We define φ_{T,t_0} as the conjunction of the following $\mathcal{ALC}$-LTL$_{|gGCI}$ formulae. First, every domain element must have exactly one tile type:

$$\Box\Box^-\Big(\top \sqsubseteq \bigsqcup_{t\in T}\Big(P_t \sqcap \prod_{t'\in T,\, t\neq t'} \neg P_{t'}\Big)\Big).$$

For the vertical dimension, we enforce an infinite rigid r-chain starting from a and restrict adjacent tile types to be compatible:

$$\Box\Box^-(\top \sqsubseteq \exists r.\top), \quad \Box\Box^-\Big(P_t \sqsubseteq \bigsqcup_{(t,t')\in V} \forall r.P_{t'}\Big).$$

For each time point i, we want to mark all individuals along the r-chain with the concept name $A^{(i \bmod 3)}$. Instead of expressing this for all domain elements, we describe the progression of the counter only for a, and then transfer the counter value to all r-connected elements, using the following formulae for all $0 \leq i \leq 2$:

$$A^0(a), \quad \Box\Box^-\big(A^i(a) \to \bigcirc A^{i+1}(a)\big), \quad \Box\Box^-\big(A^i \sqsubseteq \neg A^{i+1} \sqcap \forall r.A^i\big).$$

The last GCI states two things simultaneously: first, A^i and A^{i+1} are disjoint, and, second, the value of A^i is transferred to all r-successors.

To encode the compatibility of horizontally adjacent tiles, we add the following formulae for all $0 \leq i \leq 2$ and $t \in T$:

$$\Box\Box^-\Big(P_t \sqcap A^i \sqsubseteq \bigsqcup_{(t,t')\in H} N^i_{t'}\Big), \quad \Box\Box^-\big(N^i_t \sqcap A^{i+1} \sqsubseteq P_t\big), \quad \Box\Box^-\big(A^{i-1} \sqsubseteq \neg N^i_t\big).$$

Similar to the proof of Lemma 5.1, these express that any domain element with tile type t (expressed by P_t) at a time point marked with A^i must have a compatible type t' at the next time point (expressed by $N^i_{t'}$). Since all $N^i_{t'}$ are false at the previous time point (designated by A^{i-1}) and $\mathsf{iRig}(N^i_{t'}) = 2$, any $N^i_{t'}$ that holds at the current time point is still satisfied at the next time point (marked by A^{i+1}), where it then implies $P_{t'}$.

Finally, we express the condition on t_0 via the formula

$$\Box\Diamond P_{t_0}(a).$$

We now obtain the claimed undecidability from known results about the tiling problem [34].

Theorem 5.5. *Satisfiability in $\mathcal{ALC}$-LTL$_{|gGCI}$ with rigid roles and interval-rigid concepts is Σ^1_1-hard, and thus not recursively enumerable.*

6 $\mathcal{ALC}$-LTL$^{\text{bin}}$ WITHOUT INTERVAL-RIGID NAMES

To conclude our investigation of metric temporal DLs, we consider the setting of $\mathcal{ALC}$-LTL$^{\text{bin}}$ without interval-rigid names. Observe that all lower bounds follow from known results, since they hold already without past operators over a timeline given by $\mathbb{N}$. In particular, ExpSpace-hardness for $\mathcal{ALC}$-LTL$^{\text{bin}}_{|gGCI}$ is inherited from LTL$^{\text{bin}}$ [3, 29], while rigid role names increase the complexity to 2-ExpTime in $\mathcal{ALC}$-LTL$_{|gGCI}$ [20] (see Table 2).

The upper bounds can be shown using an existing approach [20]. The idea is to split the satisfiability test into two parts: one for the temporal and one for the DL dimension. In what follows, let φ be an $\mathcal{ALC}$-LTL$^{\text{bin}}$ formula. The *propositional abstraction* φ^{p} is the propositional LTL$^{\text{bin}}$ formula obtained from φ by replacing every $\mathcal{ALC}$ axiom by a propositional variable in such a way that there is a one-to-one relationship between the $\mathcal{ALC}$ axioms $\alpha_1, \ldots, \alpha_m$ occurring in φ and the propositional variables $p_1, \ldots, p_m$ in φ^{p}.

The general idea is to first verify the existence of a model of φ^{p}, and then to use it to construct a model of φ (if it exists). While satisfiability of φ implies that φ^{p} is also satisfiable, the converse is not true. For example, the propositional abstraction $p \wedge q \wedge \neg r$ of $\varphi = A \sqsubseteq B \wedge A(a) \wedge \neg B(a)$ is satisfiable, while φ is not. To rule out such cases, we collect the propositional worlds occurring in a model of φ^{p} into a (non-empty) set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$, which is then used to check the satisfiability of the original formula (with rigid names). This is captured by the LTL$^{\mathsf{bin}}$ formula $\varphi^{\mathrm{p}}_{\mathcal{W}} := \varphi^{\mathrm{p}} \wedge \varphi_{\mathcal{W}}$, where $\varphi_{\mathcal{W}}$ is the (exponential) LTL formula

$$\Box\Box^{-} \bigvee_{W \in \mathcal{W}} \left(\bigwedge_{p \in W} p \wedge \bigwedge_{p \in \overline{W}} \neg p \right),$$

in which $\overline{W} := \{p_1, \dots, p_m\} \setminus W$ denotes the complement of W. The formula $\varphi^{\mathrm{p}}_{\mathcal{W}}$ states that, when looking for a propositional model of φ^{p}, we are only allowed to use worlds from $\mathcal{W}$.

Since satisfiability of φ implies satisfiability of $\varphi^{\mathrm{p}}_{\mathcal{W}}$ for some $\mathcal{W}$, we can proceed as follows: we first choose a set $\mathcal{W}$ of worlds, test whether $\varphi^{\mathrm{p}}_{\mathcal{W}}$ is satisfiable, and then check whether a model with worlds from $\mathcal{W}$ can indeed be lifted to a temporal DL interpretation (respecting rigid names). To check the latter, we consider the conjunction $\bigwedge_{p_j \in W} \alpha_j \wedge \bigwedge_{p_j \in \overline{W}} \neg \alpha_j$ for every $W \in \mathcal{W}$. Note that the rigid names require that all these conjunctions are *simultaneously* checked for satisfiability. To tell apart the *flexible* names X occurring in different elements of $\mathcal{W} = \{W_1, \dots, W_k\}$, we introduce copies $X^{(i)}$ for all $i \in [1, k]$. The axioms $\alpha_j^{(i)}$ are obtained from α_j by replacing every flexible name X by $X^{(i)}$, which yields the following conjunction of exponential size:

$$\chi_{\mathcal{W}} := \bigwedge_{i=1}^{k} \Big(\bigwedge_{p_j \in W_i} \alpha_j^{(i)} \wedge \bigwedge_{p_j \in \overline{W_i}} \neg \alpha_j^{(i)} \Big).$$

The following known characterisation [20] can be easily adapted to our setting.

Lemma 6.1 ([20]). *An $\mathcal{ALC}$-LTL$^{\mathsf{bin}}$ formula φ is satisfiable with rigid names iff there is a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ such that $\varphi^{\mathrm{p}}_{\mathcal{W}}$ and $\chi_{\mathcal{W}}$ are both satisfiable.*

From this, we can easily obtain the upper bounds for $\mathcal{ALC}$-LTL$^{\mathsf{bin}}$. Note that the ExpSpace upper bound already follows from Theorem 5.3.

Theorem 6.2. *Satisfiability in $\mathcal{ALC}$-LTL$^{\mathsf{bin}}$ is in 2-ExpTime with rigid names, and in ExpSpace with rigid concepts.*

Proof. By Lemma 6.1, the satisfiability of φ can be decided using the following steps:

(1) find a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$,
(2) check the satisfiability of $\chi_{\mathcal{W}}$, and
(3) check the satisfiability of $\varphi^{\mathrm{p}}_{\mathcal{W}} = \varphi^{\mathrm{p}} \wedge \varphi_{\mathcal{W}}$.

Depending on the targeted complexity class, Step (1) can be handled in the following different ways [20]:

- one can guess a set $\mathcal{W}$ in (non-deterministic) exponential time, or
- one can enumerate all sets $\mathcal{W}$ in (deterministic) double exponential time.

Moreover, Step (2) is exactly the same as for $\mathcal{ALC}$-LTL, which means that the know complexity upper bounds [20] directly apply to Steps (1) and (2). In particular, they are possible in 2-ExpTime in general and in NExpTime if we only allow rigid concepts. It only remains to determine the complexity of Step (3), and take the union of the obtained complexity classes.

For this, we translate φ^{p} into an exponentially larger LTL formula $\varphi^{\mathrm{p}\prime}$ using Proposition 3.3. Since the exponentially large $\varphi_{\mathcal{W}}$ is already an LTL formula, the satisfiability of $\varphi^{\mathrm{p}\prime} \wedge \varphi_{\mathcal{W}}$ can be checked in exponential space. This yields the claimed results since NExpTime $\subseteq$ (N)ExpSpace $\subseteq$ 2-ExpTime. □

As a result, in most cases, the complexity of the DL part is dominated by the ExpSpace complexity of the temporal part. The only exception is the 2-ExpTime bound for $\mathcal{ALC}$-LTL$^{\mathsf{bin}}$ with rigid names.

7 DISCUSSION AND RELATED WORK

Temporal DLs have been extensively studied (see [39] for an overview). The most closely related works [16, 20, 32] were already mentioned in the introduction. Our main contribution, presented in Section 4, is a non-trivial extension of LTL$^{\mathsf{bin}}_{\mathcal{ALC}}$ [32] with interval-rigid names and assertions, and its complexity landscape. We also studied, in Sections 5 and 6, combinations of $\mathcal{ALC}$-LTL [20] with interval-rigid names and metric temporal logic, respectively. Our results are based on a previous conference paper [16], the main difference being the change of temporal semantics from $\mathbb{N}$ to $\mathbb{Z}$, the encoding of assertions into GCIs (Section 3.1), and the proof that satisfiability in $\mathcal{ALC}$-LTL with interval-rigid names is 2-ExpSpace-hard (Section 5.2). We now discuss the relation with some other works on (metric) temporal (description) logic.

Temporal Roles. A temporal operator $\Box$ applied to a single role leads to undecidability in LTL$^{\mathsf{bin}}_{\mathcal{ALC}}$, since one could then easily adapt the undecidability proof for satisfiability of LTL$_{\mathcal{ALC}}$ with rigid roles [39]. Artale et al. [11] show that one can combine $\mathcal{ALC}$ concepts with the qualitative operators $\Diamond$ and $\Box$ on roles by disallowing the $\bigcirc$-operator, but they do not consider quantitative variants.

Interval Temporal Logic. There are works combining DLs with (sublogics of) Halpern and Shoham's interval logic [4, 9, 42], which uses Allen's 13 relations between intervals to reason about the relative temporal position of different events. This setting is quite different from ours, since it uses intervals (rather than time points) as the basic time units.

Metric Temporal Logic. Extensions of propositional logic with metric temporal operators have been proposed for both linear [2, 3] and branching time [28]. In linear time, two semantics have been studied: continuous and pointwise [40]. In the pointwise semantics, a monotonic function associates each state with a time point. This mapping may not be continuous, as there may be time points not associated with any state whereas in the continuous semantics each time point is associated with a state (and vice-versa). One can polynomially translate a metric temporal logic formula under the pointwise semantics into a formula under the continuous semantics over the natural numbers [32, 35]. Once the semantics is chosen, there is the choice of timeline. In both semantics, propositional metric temporal logic over the reals is undecidable [3], but decidable fragments have been identified [1, 24]. In this work, we adopted a linear continuous semantics over the integers. Recently, an interesting metric temporal extension of Datalog over the reals was proposed, which however can express neither interval-rigid names nor existential restrictions [25, 26, 47].

Timeline. The complexity of temporal $\mathcal{ALC}$ has been extensively studied over the natural numbers [20, 30, 48] and, more recently, over finite intervals of natural numbers [12–14]. As already mentioned, our results build on our previous work [16] for a timeline given by $\mathbb{N}$. The satisfiability of formulae in the logics we consider may change depending on the chosen timeline. For instance, if A is a 2-rigid concept name, then the formula $A(a) \wedge \bigcirc\Box(\top \sqsubseteq \neg A)$ is only satisfiable over $\mathbb{Z}$, whereas the formula $B(a) \wedge \Box^{-}(\top \sqsubseteq \neg\bigcirc B)$ cannot be satisfied over $\mathbb{Z}$, but is satisfiable over $\mathbb{N}$. Regarding the quasimodel construction in Section 4.2, a central observation was that, with a semantics over $\mathbb{Z}$, every satisfiable formula has a constant quasimodel. This is not the case if the semantics is defined

for $\mathbb{N}$, since formulae of the form $\Box(\top \sqsubseteq \neg\bigcirc C)$ enforce that a concept type can only contain C at time point 0. Also, over $\mathbb{Z}$, a quasistate at time point 1 can contain role constraints $\sigma \overset{s}{k} \sigma'$ with $k > 1$, which cannot happen over $\mathbb{N}$.

Other Description Logics and Reasoning Problems. In principle, the arguments for $\mathcal{ALC}$-LTL$^{\text{bin}}$ in Section 6 are also applicable if we replace $\mathcal{ALC}$ by the lightweight DLs *DL-Lite* or $\mathcal{EL}$, yielding tight complexity bounds based on known results [7, 23]. The complexity of temporal extensions of *DL-Lite* with interval-rigid roles and metric operators has been investigated in detail [46]. Apart from satisfiability, the complexity of a variety of combinations of temporalised DLs and query languages, in particular based on *conjunctive queries*, has been investigated previously [5, 10, 17, 18, 21–23, 31]; for a recent survey, see [6]. Probabilistic extensions of temporalised conjunctive queries are also becoming the subject of investigations [36].

8 CONCLUSIONS

We have investigated a series of extensions of LTL$_{\mathcal{ALC}}$ and $\mathcal{ALC}$-LTL with interval-rigid names and metric temporal operators, with complexity results ranging from ExpTime to 2-ExpSpace. This paper provides a comprehensive guide to the complexities faced by applications that want to combine ontological reasoning with quantitative temporal logics. There are several interesting dimensions to explore further, guided by practical applications. These include using DLs other than $\mathcal{ALC}$, metric temporal extensions of conjunctive queries, combinations with concrete domains and probabilistic reasoning to model (uncertain) measurements, and using metric temporal semantics based on bounded, finite, and dense timelines. An interesting extension of interval-rigid roles would be to make the length of the interval-rigidity dependent on the role fillers. For example, a `finding` of `Influenza` will typically have a different duration than one of `ChickenPox`, so one could require any pair of elements $(d, e) \in \texttt{finding}^{\mathcal{I}_i}$ with $e \in \texttt{Influenza}^{\mathcal{I}_i}$ to keep these two properties for at least 7 days, and choose a longer duration in the case of `ChickenPox`. We conjecture that such an extension would not affect our complexity results.

ACKNOWLEDGMENTS

This work was partially supported by the DFG in the CRC 912 (HAEC), the project BA 1122/19-1 (GOASQ), the Cluster of Excellence "Center for Advancing Electronics Dresden" (cfaed), and grant 389792660 (https://www.perspicuous-computing.science/) (TRR 248, Center for Perspicuous Computing). We are very grateful to the anonymous reviewers for their suggestions to improve the paper.

A PROOFS FOR SECTION 3

Theorem 3.1. *Satisfiability of* LTL$^{\text{bin}}_{\mathcal{ALC}|gGCI}$ *formulae can be polynomially reduced to satisfiability of* LTL$^{\text{bin}}_{\mathcal{ALC}|gGCI}$ *formulae without assertions.*

Proof. We show that $\varphi \wedge \Box\Box^{-}\mathcal{T}$ is satisfiable iff $\Box\Box^{-}(\mathcal{T}' \wedge (\top \sqsubseteq \exists r_0.A_\varphi))$ is satisfiable, where r_0 is a fresh role name.

($\Leftarrow$) Let $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ be a model of $\Box\Box^{-}(\mathcal{T}' \wedge (\top \sqsubseteq \exists r_0.A_\varphi))$. Since $\mathcal{T} \subseteq \mathcal{T}'$, we have $\mathfrak{I}, 0 \models \Box\Box^{-}\mathcal{T}$. We extend $\mathfrak{I}$ to an interpretation $\mathfrak{I}' = (\Delta^{\mathfrak{I}'}, (\mathcal{I}'_i)_{i\in\mathbb{Z}})$ for which additionally $\mathfrak{I}', 0 \models \varphi$. Since $\mathfrak{I}, 0 \models \top \sqsubseteq \exists r_0.A_\varphi$ and the domain must not be empty by definition, there exists a domain element $d^\star \in \Delta^{\mathfrak{I}}$ such that $d^\star \in (A_\varphi)^{\mathcal{I}_0}$. Based on $d^\star$, we extend $\mathfrak{I}$ as follows.

- $\Delta^{\mathfrak{I}'} := \Delta^{\mathfrak{I}} \cup \{d_a \text{ fresh} \mid a \in \mathsf{N_I}(\varphi)\}$.
- For all $a \in \mathsf{N_I}(\varphi)$, set $a^{\mathfrak{I}'} := d_a$.
- For all $i \in \mathbb{Z}$ and $A \in \mathsf{N_C}$, set $A^{\mathcal{I}'_i} := A^{\mathcal{I}_i} \cup \{d_a \mid d^\star \in A^{\mathcal{I}_i}_{A(a)}\}$.

- For all $i \in \mathbb{Z}$ and $r \in \mathsf{N_R}$, set $r^{\mathcal{I}'_i} := r^{\mathcal{I}_i} \cup \{(d_a, d_b) \mid d^\star \in A^{\mathcal{I}_i}_{r(a,b)}\} \cup \{(d_a, e) \mid (d^\star, e) \in r_a^{\mathcal{I}_i}\}$.

It is easy to show by structural induction on C that $C^{\mathcal{I}'_i} \cap \Delta^{\mathfrak{I}} = C^{\mathcal{I}_i}$ for all concepts C and $i \in \mathbb{Z}$; that is, the interpretation of all concepts on the original domain remains unchanged. In particular, all GCIs in $\mathcal{T}$ remain satisfied on the elements of $\Delta^{\mathfrak{I}}$. It is also straightforward to show by structural induction that, for all assertions α over the concepts in $\mathsf{sub}^{\mathsf{c}}(\varphi) \cup \mathsf{sub}^{\mathsf{c}}(\mathcal{T})$ and the individual names $\mathsf{N_I}(\varphi)$, it holds that $\mathfrak{I}', i \models \alpha$ iff $d^\star \in A^{\mathcal{I}_i}_\alpha$, for all $i \in \mathbb{Z}$. We only describe the case of existential restrictions here in detail. The other cases can be handled similarly.

Consider an assertion of the form $(\exists r.C)(a)$. First, if $d_a \in (\exists r.C)^{\mathcal{I}'_i}$, then there is $e \in \Delta^{\mathfrak{I}'}$ such that $(d_a, e) \in r^{\mathcal{I}'_i}$ and $e \in C^{\mathcal{I}'_i}$. If $e = d_b$ for some $b \in \mathsf{N_I}(\varphi)$, then $\mathfrak{I}', i \models r(a,b) \sqcap C(b)$, $d^\star \in A^{\mathcal{I}_i}_{r(a,b)}$ and $d^\star \in A^{\mathcal{I}_i}_{C(b)}$, and because $\mathfrak{I}, i \models A_{(\exists r.C)(a)} \equiv \bigsqcup_{b \in \mathsf{N_I}(\varphi)}(A_{r(a,b)} \sqcap A_{C(b)}) \sqcup \exists r_a.C$, also $d^\star \in A^{\mathcal{I}_i}_{(\exists r.C)(a)}$. Otherwise, $e \neq d_b$ for all $b \in \mathsf{N_I}(\varphi)$, which implies $(d^\star, e) \in r_a^{\mathcal{I}_i}$ by the definition of $\mathfrak{I}'$. Since $e \in C^{\mathcal{I}'_i}$ and $e \in \Delta^{\mathfrak{I}}$, also $e \in C^{\mathcal{I}_i}$. But then, $d^\star \in (\exists r_a.C)^{\mathcal{I}_i}$, which together with our equivalence axiom implies $d^\star \in A^{\mathcal{I}_i}_{(\exists r.C)(a)}$. We obtain that $\mathfrak{I}', i \models (\exists r.C)(a)$ implies $d^\star \in A^{\mathcal{I}_i}_{(\exists r.C)(a)}$.

Now assume that $d^\star \in A^{\mathcal{I}_i}_{(\exists r.C)(a)}$. Because $\mathfrak{I}, i \models A_{(\exists r.C)(a)} \equiv \bigsqcup_{b \in \mathsf{N_I}(\varphi)}(A_{r(a,b)} \sqcap A_{C(b)}) \sqcup \exists r_a.C$, either i) for some $b \in \mathsf{N_I}(b)$, we have $d^\star \in A^{\mathcal{I}_i}_{r(a,b)}$ and $d^\star \in A_{C(b)}$, or ii) $d^\star \in (\exists r_a.C)^{\mathcal{I}_i}$. In both cases, it follows by our inductive hypothesis and by construction that $d_a \in (\exists r.C)^{\mathcal{I}'_i}$ and $\mathfrak{I}', i \models (\exists r.C)(a)$.

This concludes our induction, which also shows that the GCIs $C \sqsubseteq D$ from $\mathcal{T}$ are satisfied by the new domain elements of the form d_a, due to the axioms $A_{C(a)} \sqsubseteq A_{D(a)}$ in $\mathcal{T}'$, which are satisfied by $\mathfrak{I}$. A similar induction over the subformulae of φ can be used to show that $\mathfrak{I}', 0 \models \varphi$. As a consequence, we obtain $\mathfrak{I}', 0 \models \varphi \wedge \Box\Box^-\mathcal{T}$.

($\Rightarrow$) Let $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i \in \mathbb{Z}})$ be a model of $\varphi \wedge \Box\Box^-\mathcal{T}$. Based on $\mathfrak{I}$, we construct a model $\mathfrak{I}'$ of $\Box\Box^-(\mathcal{T}' \wedge (\top \sqsubseteq \exists r_0.A_\varphi))$ as follows, where $\mathfrak{I}' = (\Delta^{\mathfrak{I}'}, (\mathcal{I}'_i)_{i \in \mathbb{Z}})$. Note that $\top \sqsubseteq \exists r_0.A_\varphi$ requires us to model the satisfaction of φ not only at time point 0, but actually at every time point. To this end, the domain of the new interpretation basically consists of multiple copies of $\Delta^{\mathfrak{I}}$, regarded as fresh at each time point. The model $\mathfrak{I}'$ is now defined as follows.

(a) $\Delta^{\mathfrak{I}'} := \{d_i \mid d \in \Delta^{\mathfrak{I}}, i \in \mathbb{Z}\}$.
(b) For every $i, j \in \mathbb{Z}$ and $A \in \mathsf{N_C}(\varphi)$, set $A^{\mathcal{I}'_i} := \{d_j \mid j \in \mathbb{Z}, d \in A^{\mathcal{I}_{i+j}}\}$.
(c) For every $i, j \in \mathbb{Z}$ and $r \in \mathsf{N_R}(\varphi)$, set $r^{\mathcal{I}'_i} := \{(d_j, e_j) \mid j \in \mathbb{Z}, (d, e) \in r^{\mathcal{I}_{i+j}}\}$.
(d) For every $i \in \mathbb{Z}$, set $r_0^{\mathcal{I}_i} := \{(d, e_i) \mid d \in \Delta^{\mathfrak{I}'}, e \in \Delta^{\mathfrak{I}}\}$.
(e) For every $i, j \in \mathbb{Z}$ and concept name A_ψ, set $A_\psi^{\mathcal{I}'_i} := \{d_j \mid d \in \Delta^{\mathfrak{I}}, j \in \mathbb{Z},\ \mathfrak{I}, i + j \models \psi\}$.
(f) For every $i, j \in \mathbb{Z}$, $a \in \mathsf{N_I}(\varphi)$, and $r \in \mathsf{N_R}$, set $r_a^{\mathcal{I}'_i} := \{(d_j, e_j) \mid d \in \Delta^{\mathfrak{I}}, j \in \mathbb{Z}, (a^{\mathcal{I}}, e) \in r^{\mathcal{I}_{i+j}}\}$.

Items (a)–(c) ensure that, for each $i \in \mathbb{Z}$, $\mathfrak{I}'$ contains a copy of interpretation $\mathfrak{I}$ shifted by i time units; for instance, if $e \in A^{\mathcal{I}_5}$, then, by definition, $e_6 \in A^{\mathcal{I}'_{-1}}$, $e_5 \in A^{\mathcal{I}'_0}$, $e_4 \in A^{\mathcal{I}'_1}$, $e_3 \in A^{\mathcal{I}'_2}$, etc. This yields $\mathfrak{I}', 0 \models \Box\Box^-\mathcal{T}$, because the remaining items refer to names that do not occur in $\mathcal{T}$. Additionally, all domain elements encode the timeline of the assertions, where again each d_i represents the assertions that are to be satisfied shifted by i time points. The GCI $\top \sqsubseteq \exists r_0.A_\varphi$ can thus be satisfied at every time point, which is established by Items (d) and (e). Finally, given the semantics of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$, Items (e) and (f) yield that the remaining axioms in $\mathcal{T}'$ are also satisfied at every time point, because no other item refers to the freshly introduced concept and role names. Hence, we obtain $\mathfrak{I}', 0 \models \Box\Box^-(\mathcal{T}' \wedge (\top \sqsubseteq \exists r_0.A_\varphi))$. □

Theorem 3.2. *Satisfiability of* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae can be polynomially reduced to satisfiability of* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae without assertions.*

Proof. We show that φ is satisfiable iff $\varphi' \wedge \varphi_{\mathcal{A}} \wedge \Box\Box^-\mathcal{T}'$ is satisfiable.

($\Rightarrow$) Assume that φ is satisfiable, and let $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ be a model of φ. We extend $\mathfrak{I}$ to a model $\mathfrak{I}' = (\Delta^{\mathfrak{I}}, (\mathcal{I}'_i)_{i\in\mathbb{Z}})$ of $\varphi' \wedge \varphi_{\mathcal{A}} \wedge \Box\Box^{-}\mathcal{T}'$. To this end, for every $i \in \mathbb{Z}$, we extend $\mathcal{I}_i$ to $\mathcal{I}'_i$ as follows.

(1) For every $X \in \mathsf{N_C}(\varphi) \cup \mathsf{N_R}(\varphi)$, set $X^{\mathcal{I}'_i} := X^{\mathcal{I}_i}$.
(2) For every introduced concept name A_α, set $A_\alpha^{\mathcal{I}'_i} := \Delta^{\mathfrak{I}}$ if $\mathfrak{I}, i \models \alpha$, and otherwise $A_\alpha^{\mathcal{I}'_i} := \emptyset$.
(3) For every $r \in \mathsf{N_R}(\varphi)$ and $a \in \mathsf{N_I}(\varphi)$, set $r_a^{\mathcal{I}'_i} := \{(d, e) \mid d \in \Delta^{\mathfrak{I}},\ (a^{\mathfrak{I}}, e) \in r^{\mathcal{I}_i}\}$.

Note that r_a again captures the r-successors of a, which are now reachable by all domain elements, since the concept names A_α encoding the assertional behaviour are now satisfied equally by all domain elements, or by none. Since the interpretation of the names in $\mathsf{N_C}(\varphi) \cup \mathsf{N_R}(\varphi)$ remains unchanged, for every GCI $C \sqsubseteq D$ occurring in φ and $i \in \mathbb{Z}$, we have $\mathfrak{I}', i \models C \sqsubseteq D$ iff $\mathfrak{I}, i \models C \sqsubseteq D$. Item (2) yields that, for all $i \in \mathbb{Z}$ and assertions α occurring in φ, $\mathfrak{I}', i \models \top \equiv A_\alpha$ iff $\mathfrak{I}, i \models \alpha$. Since $\mathfrak{I}, 0 \models \varphi$, a standard structural induction yields that $\mathfrak{I}', 0 \models \varphi'$. We now consider any of the formulae

$$\Box\Box^{-}\Big((C \sqsubseteq D) \rightarrow \bigwedge_{a\in\mathsf{N_I}(\varphi)} \big(A_{C(a)} \sqsubseteq A_{D(a)}\big)\Big)$$

in $\varphi_{\mathcal{A}}$. Assume that $\mathfrak{I}', i \models C \sqsubseteq D$ for any $i \in \mathbb{Z}$, and let $a \in \mathsf{N_I}(\varphi)$. Since both $A_{C(a)}$ and $A_{D(a)}$ are equivalent to either $\top$ or $\bot$ in $\mathcal{I}'_i$, the only interesting case is where $\mathfrak{I}', i \models \top \equiv A_{C(a)}$. By the arguments above, we obtain $\mathfrak{I}, i \models C \sqsubseteq D$ as well as $\mathfrak{I}, i \models C(a)$, and hence $\mathfrak{I}, i \models D(a)$, which again yields $\mathfrak{I}', i \models \top \equiv A_{D(a)}$, and finally $\mathfrak{I}', i \models A_{C(a)} \sqsubseteq A_{D(a)}$.

The remaining axioms in $\varphi_{\mathcal{A}}$ of the form $\Box\Box^{-}\big((\top \equiv A_\alpha) \vee (\bot \equiv A_\alpha)\big)$ are clearly also satisfied. Finally, $\mathfrak{I}', 0 \models \Box\Box^{-}\mathcal{T}'$ follows by comparing the axioms to the semantic conditions of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$.

($\Leftarrow$) Let $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ be a model of $\varphi' \wedge \varphi_{\mathcal{A}} \wedge \Box\Box^{-}\mathcal{T}'$. We choose an arbitrary individual $d^\star \in \Delta^{\mathfrak{I}}$ and adapt $\mathfrak{I}$ to a model $\mathfrak{I}' = (\Delta^{\mathfrak{I}'}, (\mathcal{I}'_i)_{i\in\mathbb{Z}})$ of φ as follows.

(1) $\Delta^{\mathfrak{I}'} := \Delta^{\mathfrak{I}} \cup \{d_a \text{ fresh} \mid a \in \mathsf{N_I}(\varphi)\}$.
(2) For every $a \in \mathsf{N_I}(\varphi)$, set $a^{\mathfrak{I}'} := d_a$.
(3) For every $i \in \mathbb{Z}$, $A \in \mathsf{N_C}$, and $a \in \mathsf{N_I}(\varphi)$, set $A^{\mathcal{I}'_i} := A^{\mathcal{I}_i} \cup \{d_a \mid d^\star \in A^{\mathcal{I}}_{A(a)}\}$.
(4) For every $i \in \mathbb{Z}$ and $r \in \mathsf{N_C}$, set $r^{\mathcal{I}'_i} = r^{\mathcal{I}_i} \cup \{(d_a, d_b) \mid d^\star \in A^{\mathcal{I}_i}_{r(a,b)}\} \cup \{(d_a, e) \mid (d^\star, e) \in r_a^{\mathcal{I}_i}\}$.

Since $\mathfrak{I}, 0 \models \varphi_{\mathcal{A}}$, all domain elements must agree on the concept names A_α, and hence which d we choose is irrelevant, and we can show similarly as in the proof of Lemma 3.1 that $\mathfrak{I}', i \models \alpha$ iff $d^\star \in A_\alpha^{\mathcal{I}_i}$ iff $A_\alpha^{\mathcal{I}_i} = \Delta^{\mathfrak{I}}$ iff $\mathfrak{I}, i \models \top \equiv A_\alpha$, for all relevant assertions α. Since $\mathfrak{I}, 0 \models \varphi'$, it only remains to show that the same holds for GCIs, i.e., we have $\mathfrak{I}', i \models C \sqsubseteq D$ iff $\mathfrak{I}, i \models C \sqsubseteq D$, and the claim then follows by straightforward structural induction. Again as in the proof of Lemma 3.1, $C^{\mathcal{I}'_i} \cap \Delta^{\mathfrak{I}} = C^{\mathcal{I}_i}$ can be shown for all concepts C. Hence, if $\mathfrak{I}', i \models C \sqsubseteq D$, then we immediately have $\mathfrak{I}, i \models C \sqsubseteq D$. On the other hand, if $\mathfrak{I}, i \models C \sqsubseteq D$, then because of $\mathfrak{I}, 0 \models \varphi_{\mathcal{A}}$ the GCI is also satisfied on the new domain elements. □

Theorem 3.4. *Every* $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ *formula can be translated in polynomial time into an equisatisfiable* $\mathrm{LTL}_{\mathcal{ALC}}$ *formula.*

Proof. We first describe the reduction of temporal concepts. As already mentioned, the basic idea [38] is to use a counter to determine the distance to the *nearest* time point that makes a concept of the form $C\mathcal{U}_{[0,c]}D$ true (i.e., it satisfies D, and C is satisfied at all time points in between). This counter is initialised whenever D is satisfied, counts (backwards in time) from 0 up to $c+1$ as long as C is satisfied, and then stays at $c+1$. The concept $C\mathcal{U}_{[0,c]}D$ is satisfied iff the counter value is $\leq c$. The counter is represented by fresh concept names, one for each bit in the binary representation, of which there are polynomially many. This suffices since, for each individual and each time point,

there is a unique nearest time point satisfying D, which uniquely determines the counter value. For a concept of the form $C\,\mathcal{U}_{[c,\infty)}D$, we can use a similar counter, which however counts the distance to the *furthest* occurrence of D that makes $C\,\mathcal{U}_{[c,\infty]}D$ true. Then, $C\,\mathcal{U}_{[c,\infty)}D$ is satisfied iff the counter value is $\geq c$. The idea for $C\,\mathcal{S}_{[0,c]}D$ and $C\,\mathcal{S}_{[c,\infty)}D$ is the same, but the counter is increased in the other direction of the timeline.

More formally, let φ be an $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ formula. For each subconcept of the form $F = C\,\mathcal{U}_I D$ or $F = C\,\mathcal{S}_I D$ occurring in φ, where I is either $[0,c]$ or $[c,\infty)$, we introduce fresh concept names $A^F_0, \ldots, A^F_{\ell_F}$, where $\ell_F = \lceil \log(c+1) \rceil - 1$. Then, concepts such as $\big(A^F \leq c\big)$ are used to compare the counter value represented by the concept names A^F_i to the value of c. More precisely, we define the following abbreviations, where for any $d \in [0, \ldots, 2^{\ell_F})$ and $0 \leq i \leq \ell_F$, we have that d_i denotes the i-th bit of the binary representation of d.

$$\big(A^F_i = d_i\big) \coloneqq \begin{cases} A^F_i & \text{if } d_i = 1 \\ \neg A^F_i & \text{otherwise} \end{cases} \qquad \big(A^F_i < d_i\big) \coloneqq \begin{cases} \neg A^F_i & \text{if } d_i = 1 \\ \bot & \text{otherwise} \end{cases}$$

$$\big(A^F = d\big) \coloneqq \prod_{i=0}^{\ell_F} \big(A^F_i = d_i\big) \qquad \big(A^F < d\big) \coloneqq \bigsqcup_{i=0}^{\ell_F} \left(\big(A^F_i < d_i\big) \sqcap \prod_{j=i+1}^{\ell_F} \big(A^F_i = d_i\big) \right)$$

$$\big(A^F \leq d\big) \coloneqq \big(A^F = d\big) \sqcup \big(A^F < d\big) \qquad \big(A^F > d\big) \coloneqq \neg\big(A^F \leq d\big) \qquad \big(A^F \geq d\big) \coloneqq \neg\big(A^F < d\big)$$

To express that the counter value is decremented along the temporal dimension, we use the concept

$$(A^F{-}{-}) \coloneqq \prod_{i=0}^{\ell_F} \left(\left(\prod_{j=0}^{i-1} \bigcirc A^F_j \right) \leftrightarrow \Big(\big(\bigcirc A^F_i\big) \leftrightarrow \big(\neg A^F_i\big) \Big) \right),$$

which flips the i-th bit of A^F iff all lower bits at the next time point are set. We now define a recursive transformation $\cdot^*$ on $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ concepts as follows.

$$A^* \coloneqq A \quad \text{for all } A \in \mathsf{N_C} \cup \{\top\} \qquad (\neg C)^* \coloneqq \neg C^*$$

$$(\exists r.C)^* \coloneqq \exists r.C^* \qquad (C \sqcap D)^* \coloneqq C^* \sqcap D^*$$

$$(\bigcirc C)^* \coloneqq \bigcirc C^* \qquad (\bigcirc^- C)^* \coloneqq \bigcirc^- C^*$$

$$(C\,\mathcal{U}_{[0,c]}D)^* \coloneqq (C^*\,\mathcal{U}\,D^*) \sqcap \big(A^{C\,\mathcal{U}_{[0,c]}D} \leq c\big) \qquad (C\,\mathcal{U}_{[c,\infty)}D)^* \coloneqq (C^*\,\mathcal{U}\,D^*) \sqcap \big(A^{C\,\mathcal{U}_{[c,\infty)}D} \geq c\big)$$

$$(C\,\mathcal{S}_{[0,c]}D)^* \coloneqq (C^*\,\mathcal{S}\,D^*) \sqcap \big(A^{C\,\mathcal{S}_{[0,c]}D} \leq c\big) \qquad (C\,\mathcal{S}_{[c,\infty)}D)^* \coloneqq (C^*\,\mathcal{S}\,D^*) \sqcap \big(A^{C\,\mathcal{S}_{[c,\infty)}D} \geq c\big)$$

The concepts $C^*\mathcal{U}D^*$ and $C^*\mathcal{S}D^*$ ensure that the counter value is well-defined, i.e., that there actually exists a future or past time point that satisfies D and C is satisfied in the meantime. Finally, we obtain φ^* from φ by replacing every concept C by C^*.

To describe the behaviour of the counter, we use the global GCI $\top \sqsubseteq B^F_0 \sqcap B^F_1 \sqcap B^F_2$, where for $F = C\,\mathcal{U}_{[0,c]}D$, the three concepts are defined as follows.

$$B^F_0 \coloneqq D^* \leftrightarrow \big(A^F = 0\big)$$

$$B^F_1 \coloneqq \big(C^* \sqcap \neg D^* \sqcap \bigcirc\big(A^F \leq c\big)\big) \rightarrow (A^F{-}{-})$$

$$B^F_2 \coloneqq \big(C^* \sqcap \neg D^* \sqcap \bigcirc\big(A^F = c+1\big)\big) \rightarrow \big(A^F = c+1\big)$$

These concepts express that the counter A^F is reset to 0 whenever we encounter D (enforced by B^F_0), it is increased (backwards in time) up to $c+1$ as long as C remains satisfied (B^F_1), and it then remains at $c+1$ until we encounter a previous occurrence of D, or C is not satisfied anymore (B^F_2). We thus formalise the intuition described in the beginning of this proof, that is, that the counter A^F represents the (unique) amount of time that has to elapse until the nearest occurrence of D such

that C is satisfied in the meantime. Thus, instead of $F = C\,\mathcal{U}_{[0,c]}D$, we can thus equivalently use the concept $F^* = (C^*\mathcal{U}D^*) \sqcap \big(A^F \leq c\big)$, which checks whether the counter value does not exceed c. For $F = C\,\mathcal{S}_{[0,c]}D$, we define B_0^F, B_1^F and B_2^F accordingly, where we replace each $\bigcirc$ with $\bigcirc^-$ (also within $(A^F{--})$).

For $F = C\,\mathcal{U}_{[c,\infty)}D$, we define

$$B_0^F := \big(D^* \sqcap \bigcirc\neg(C^*\mathcal{U}D^*)\big) \leftrightarrow \big(A^F = 0\big),$$
$$B_1^F := \big(C^* \sqcap \bigcirc\big(A^F \leq c\big)\big) \rightarrow (A^F{--}),$$
$$B_2^F := \big(C^* \sqcap \bigcirc\big(A^F = c+1\big)\big) \rightarrow \big(A^F = c+1\big).$$

The differences to the previous case are that the counter is only reset to 0 at the last possible D, i.e., whenever $C\mathcal{U}D$ does not hold afterwards, and that the counter is increased regardless of D. This again reflects the intuition described above, that A^F marks the distance to the last possible occurrence of D that makes F true. Intermediate occurrences of D can simply be ignored, as long as C remains satisfied. Again, for $F = C\,\mathcal{S}_{[c,\infty]}D$, we have the same definitions only with $\bigcirc$ replaced by $\bigcirc^-$ and $\mathcal{U}$ replaced by $\mathcal{S}$.

It is easy to check that φ is satisfiable iff $\varphi^* \wedge \bigwedge_F \Box\Box^-\big(\top \sqsubseteq B_0^F \sqcap B_1^F \sqcap B_2^F\big)$ is satisfiable, where F ranges over all $\mathcal{U}$- and $\mathcal{S}$-concepts in φ that are not already of the form $C\mathcal{U}D$ or $C\,\mathcal{S}D$.

To simulate $\mathcal{U}$- and $\mathcal{S}$-*formulae* ψ in φ, we use a very similar construction, where we replace concepts by formulae, i.e.,

- we define the translation $\cdot^*$ similarly on formulae,
- instead of concept names we have axioms,
- instead of $\sqcap$ we use $\wedge$,
- instead of A_i^F we use the assertion $A_i^\psi(a)$, where a is a fresh individual name,
- instead of $\top \sqsubseteq B_0^F \sqcap B_1^F \sqcap B_2^F$ we use $B_0^\psi \wedge B_1^\psi \wedge B_2^\psi$.

Overall, this yields an $\mathrm{LTL}_{\mathcal{ALC}}$ formula of polynomial size that is satisfiable iff φ is satisfiable.

To apply this construction to sublogics of $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$, observe that we only introduce global GCIs, and hence satisfiability in $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}|gGCI}$ can be polynomially reduced to satisfiability in $\mathrm{LTL}_{\mathcal{ALC}|gGCI}$. Moreover, the reduction is not affected by the rigidity of the used names, and temporal operators on the concept level are only used in the translation if they are present in φ. □

To prove Lemma 3.5, we first show the following basic result.

Lemma A.1. *An* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formula* φ *without assertions is satisfiable iff there is a quasimodel for* φ.

Proof. ($\Rightarrow$) Given an interpretation $\mathfrak{I} = (\Delta, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ such that $\mathfrak{I}, 0 \models \varphi$, we define quasistates $S(i)$ for all $i \in \mathbb{Z}$ and a corresponding set $\mathfrak{R}$ of runs as follows:

$$S(i) := \{t_{\mathcal{I}_i}(e) \mid e \in \Delta^{\mathfrak{I}}\} \cup \{\{\psi \in \mathrm{cl}^{\mathsf{f}}(\varphi) \mid \mathcal{I}_i \models \psi\}\},$$
$$\mathfrak{R} := \{(t_{\mathcal{I}_i}(e))_{i\in\mathbb{Z}} \mid e \in \Delta^{\mathfrak{I}}\},$$

where $t_{\mathcal{I}_i}(e) := \{C \in \mathrm{cl}^{\mathsf{c}}(\varphi) \mid e \in C^{\mathcal{I}_i}\}$. Taking the above definitions into account, it can be readily checked that every $S(i)$ indeed represents a quasistate, $\mathfrak{R}$ is a set of runs, and $(S, \mathfrak{R})$ is a quasimodel for φ.

($\Leftarrow$) Let $(S, \mathfrak{R})$ be a quasimodel for φ. Let $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ be an $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ interpretation, where

$$\Delta^{\mathfrak{I}} := \{d_r \mid r \in \mathfrak{R} \text{ is a concept run}\},$$
$$A^{\mathcal{I}_i} := \{d_r \mid A \in r(i),\ r \in \mathfrak{R}\},$$
$$s^{\mathcal{I}_i} := \{(d_r, d_{r'}) \mid \exists s.C \in r(i), C \in r'(i), \{\neg E \mid \neg\exists s.E \in r(i)\} \subseteq r'(i)\}.$$

By structural induction and based on the shape of quasistates and runs in quasimodels, it is easy to show that, for all runs $r \in \mathfrak{R}$, concepts $C \in \mathrm{cl}^{\mathrm{c}}(\varphi)$, and $i \in \mathbb{Z}$, we have $C \in r(i)$ iff $d_r \in C^{\mathcal{I}_i}$. Using the above conditions for quasistates and runs, **M2**, **M3**, and structural induction on formulae, one can show that, for all $i \in \mathbb{Z}$ and $\psi \in \mathrm{cl}^{\mathrm{f}}(\varphi)$, it holds that $\mathfrak{I}, i \models \psi$ iff $\psi \in t_{S(i)}$, where $t_{S(i)}$ is the formula type in $S(i)$. Hence, by **M1**, $\mathfrak{I}, 0 \models \varphi$. □

Lemma 3.5. *An* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ *formula* φ *is satisfiable iff it has a quasimodel* $(S, \mathfrak{R})$ *in which* S *is of the form*

$$^{\omega}\big(S(-k_1)\dots S(-k_2-1)\big)\, S(-k_2)\dots S(0)\dots S(k_3)\,\big(S(k_3+1)\dots S(k_4)\big)^{\omega},$$

where k_1, k_2, k_3 *and* k_4 *are bounded double exponentially in the size of* φ.

Proof. There can be at most exponentially many types for φ, and consequently at most double exponentially many quasistates. One can use an argument similar to the one that has been used for monodic first-order temporal logic, $\mathcal{QTL}_{\mathcal{U}\Box}$ [30, Theorem 11.30]. Since here we deal with the timeline $\mathbb{Z}$, the regularity needs to be present in both directions, which can easily be proven as in the proof of $\mathcal{QTL}_{\mathcal{U}\Box}$, except that we need to apply the argument in both directions. □

Lemma 3.6. *An* $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$ *formula* $\Box\Box^{-}\mathcal{T}$ *is satisfiable iff it has a constant quasimodel.*

Proof. We consider the non-trivial direction ($\Rightarrow$). Assume that there exists a quasimodel $(S, \mathfrak{R})$ for $\Box\Box^{-}\mathcal{T}$. We modify $S(n)$ by adding to $S(n)$ all concept types in $S(i)$, for all $i \in \mathbb{Z}$ with $i \neq n$. We also add to $\mathfrak{R}$ all runs in

$$\{r^{\rightarrow i} \mid r \in \mathfrak{R} \text{ a concept run}, i \in \mathbb{Z}\},$$

where $r^{\rightarrow i}(n) \coloneqq r(n+i)$ for all $r \in \mathfrak{R}$ and $n \in \mathbb{Z}$. That is, now S is a sequence with only one set of types. Since we had a single formula type in $(S, \mathfrak{R})$, by the definition of quasimodels and the fact that the formula is of the form $\Box\Box^{-}\mathcal{T}$, this set is in fact a quasistate satisfying Conditions **S1**–**S3**. We modify $\mathfrak{R}$ by adding runs which are the result of shifting the runs in $\mathfrak{R}$. This does not change the fact that they satisfy Conditions **R1**–**R4**. Finally, one easily checks that our modified $(S, \mathfrak{R})$ satisfies Conditions **M1**–**M3** and thus is a constant quasimodel for $\Box\Box^{-}\mathcal{T}$. □

To prove Theorem 3.7, we need to show that satisfiability of $\mathrm{LTL}_{\mathcal{ALC}}$ and $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}|gGCI}$ formulae is ExpSpace-complete (Lemmas A.2 and A.5); and satisfiability of $\mathrm{LTL}_{\mathcal{ALC}|gGCI}$ formulae is ExpTime-complete (Lemma A.5). By Theorems 3.2 and 3.1, we can assume w.l.o.g. that formulae do not contain assertions.

Lemma A.2. *Satisfiability in* $\mathrm{LTL}_{\mathcal{ALC}}$ *is ExpSpace-complete.*

Proof. The ExpSpace lower bound follows from known results [30, 39], since without past operators a formula is satisfiable over $\mathbb{N}$ iff it is satisfiable over $\mathbb{Z}$. We prove the upper bound. By Lemma 3.5, we only need to check for the existence of a regular quasimodel. We can use an NExpSpace algorithm like the one for $\mathcal{QTL}_{\mathcal{U}\Box}$ [30, Theorem 11.30], adapted so that it checks for the existence of a quasimodel which is regular in both directions. □

We now describe a decision procedure for checking satisfiability of $\mathrm{LTL}^{\mathrm{bin}}_{\mathcal{ALC}}$ formulae of the form $\Box\Box^{-}\mathcal{T}$, i.e., for detecting the existence of a constant quasimodel (cf. Lemma 3.6): first we find a single quasistate Q that satisfies **M1** and then determine whether, for every concept type $t \in Q$, there exists a run $r\colon \mathbb{Z} \to Q$, thus verifying Condition **M3**. Condition **M2** is satisfied trivially. The next lemmas show that this can be decided using only exponential space, only exponential time when we restrict $\mathcal{T}$ to $\mathrm{LTL}_{\mathcal{ALC}}$.

Lemma A.3. *Let Q be a quasistate for $\Box\Box^-\mathcal{T}$ and $t \in Q$. Then there exists a run $r\colon \mathbb{Z} \to Q$ with $r(0) = t$ iff there exists such a run of the following form:*

$$^{\omega}\big(r(-k_1)\dots r(-k_2-1)\big)\, r(-k_2)\dots r(0)\dots r(k_3)\,\big(r(k_3+1)\dots r(k_4)\big)^{\omega},$$

where k_1, k_2, k_3 and k_4 are double exponentially bounded in the size of $\Box\Box^-\mathcal{T}$. Moreover, if $\Box\Box^-\mathcal{T}$ contains only $\mathrm{LTL}_{\mathcal{ALC}}$ concepts, then k_1, k_2, k_3 and k_4 are single exponentially bounded.

Proof. The argument is based on the argument for PSpace-membership of satisfiability of propositional LTL formulae [45]. We are going to use the following claim.

Claim. *If r is a run $r\colon \mathbb{Z} \to Q$ such that $r^{[s,s+\ell_\mathcal{T}]} = r^{[t,t+\ell_\mathcal{T}]}$ for some $s, t \in \mathbb{N}$ with $s < t$, where $\ell_\mathcal{T}$ is the largest number occurring in any interval in $\mathcal{T}$ (or $\ell_\mathcal{T} = 1$ if no number occurs), then $r' = r^{\leq s} \cdot r^{>t}$ is also a run.*

Proof of the claim. We show that r' satisfies Conditions **R1**–**R4**. Conditions **R1** and **R2** focus on concepts of the form $\bigcirc C$ and $\bigcirc^- C$ and, especially regarding $r'(s)$ and $r'(s+1)$, they hold by the fact that $r(s+1) = r(t+1)$. Concerning Condition **R3**, which regards concepts of the form $C\,\mathcal{U}_I D$, we distinguish between whether I is of the form $[c_1, c_2]$ or $[c_1, \infty)$. In the first case, we note that, since $c_2 \leq \ell_\mathcal{T}$ any concept of this form that occurs before $r(s)$ in r has to be realised in $r^{\leq s+\ell_\mathcal{T}}$, and since $r^{[s,s+\ell_\mathcal{T}]} = r^{[t,t+\ell_\mathcal{T}]}$, it is also realised in r'. For concepts of the form $C\,\mathcal{U}_{[c_1,\infty]} D$, if they occur before $r(s)$ and are realised after $r(s+\ell_\mathcal{T})$, then we have $C\,\mathcal{U}_{[c_1,\infty]} D \in r(s+\ell_\mathcal{T}) = r(t+\ell_\mathcal{T})$, which means that they are realised again after $r(t+\ell_\mathcal{T})$, and consequently also in r'. The argument for Condition **R4** is accordingly. □

By the claim, we can eliminate any repetition of a sequence of length $\ell_\mathcal{T}$ in a run. Now, let r be any run and $0 < k_3' < k_4'$ be two integers such that $r^{[k_3'-\ell_\mathcal{T},k_3']} = r^{[k_4'-\ell_\mathcal{T},k_4']}$ and

for every $C\,\mathcal{U} D \in r(k_3')$, there is $i \in [k_3', k_4']$ such that $D \in r(i)$ and $C \in r(j)$ for all $j \in [k_3', i]$. $(*)$

It already follows that the infinite sequence

$$\dots r(0)\dots r(k_3')\,\big(r(k_3'+1)\dots r(k_4')\big)^{\omega}$$

is also a run. By Claim, we can remove any repeating sequences of length $\ell_\mathcal{T}$ from $r^{[0,k_3']}$ and $r^{(k_3',k_4']}$ such that $(*)$ does not get invalidated. Denote the resulting sequence by r' and the resulting new integers by k_3 and k_4 (as in the statement of the lemma). We give upper bounds on k_3 and k_4. First, $r'^{[0,k_3]}$ contains no sequence of length $\ell_\mathcal{T}$ twice, and since the number of types is bounded by $2^{|\varphi|}$, we obtain $k_3 \leq 2^{|\varphi|\cdot\ell_\mathcal{T}}$, which is a double exponential bound on φ (assuming binary encoding for $\ell_\mathcal{T}$), and a single exponential bound if $\ell_\mathcal{T} = 1$, which is the case if φ contains only $\mathrm{LTL}_{\mathcal{ALC}}$ concepts. Now consider k_4. For any $i_1, i_2 \in [k_3, k_4]$, such that $i_1 \leq i_2$ and $r'^{[i_1-\ell_\mathcal{T},i_1]} = r'^{[i_2-\ell_\mathcal{T},i_2]}$, there must exist some $C\,\mathcal{U} D \in r'(k_3)$ such that, for some $j \in [i_1, i_2]$, $D \in r'(j)$, since otherwise, we could have applied Claim on $r'^{[i_1-\ell_\mathcal{T},i_1]} = r'^{[i_2-\ell_\mathcal{T},i_2]}$ without invalidating $(*)$. As there are at most $|\varphi|$ different concepts of the form $C\,\mathcal{U}_I D$ in φ, it follows that every subsequence of r' of length $\ell_\mathcal{T}$ can be repeated at most $|\varphi|$ times in $r'^{[k_3-\ell_\mathcal{T},k_4]}$. As there are at most $2^{|\varphi|}$ different possible types and at most $2^{|\varphi|\cdot\ell_\mathcal{T}}$ different type sequences of this length, it follows that $k_4 \leq k_3 + 2^{|\varphi|\cdot\ell_\mathcal{T}} \cdot |\varphi|$, and thus k_4 is also double exponentially bounded in $|\varphi|$, and single exponentially bounded if φ is in $\mathrm{LTL}_{\mathcal{ALC}}$. It follows that there exists a run of the following form through t:

$$\dots r'(0)\dots r'(k_3)\,\big(r'(k_3+1)\dots r'(k_4)\big)^{\omega},$$

where k_3 and k_4 are double exponentially bounded, and single exponentially bounded if $\Box\Box^-\mathcal{T}$ contains only $\mathrm{LTL}_{\mathcal{ALC}}$ concepts. We can apply the same argument regarding $C\,\mathcal{S}_I D$ and on the subsequence $r^{(-\infty,0]}$ to show that there is also a run which is regular in both directions, and of the shape required by the lemma. □

Lemma A.4. *Given a quasistate Q for $\Box\Box^-\mathcal{T}$ and $t \in Q$, it can be decided in exponential space w.r.t. the size of $\mathcal{T}$ whether there exists a run $r : \mathbb{Z} \to Q$ such that $r(0) = t$. If $\mathcal{T}$ contains only* $\mathrm{LTL}_{\mathcal{ALC}}$ *concepts, it can be decided in exponential time in the size of $\mathcal{T}$.*

Proof. By Lemma A.3, it suffices to determine the existence of such a run of the form

$$^{\omega}\big(r(-k_1)\dots r(-k_2-1)\big)\, r(-k_2)\dots r(0)\dots r(k_3)\,\big(r(k_3+1)\dots r(k_4)\big)^{\omega}$$

with k_1, k_2, k_3 and k_4 double exponentially bounded in the size of $\mathcal{T}$.

We verify the existence of this run by considering both directions from $r(0)$, first showing the existence of a sequence $r(0)\dots r(k_3)\,\big(r(k_3+1)\dots r(k_4)\big)^{\omega}$, and afterwards the existence of a sequence $^{\omega}\big(r(-k_1)\dots r(-k_2-1)\big)\, r(-k_2)\dots r(0)$. In order to verify in non-deterministic exponential space the existence of the path $r(0)\dots r(k_3)\,\big(r(k_3+1)\dots r(k_4)\big)^{\omega}$, we first guess the numbers k_3 and k_4, which both can be stored in exponential space, and then each type $r(i)$, one after the other, keeping a window of $\ell_{\mathcal{T}}$ consecutive types in memory at each point, where $\ell_{\mathcal{T}}$ is the maximal number occurring in $\mathcal{T}$, and verifying Conditions **R1**–**R4** in each step. When we reach the window $r^{[k_3-\ell_{\mathcal{T}},k_3]}$, we additionally store it in memory, so that we can verify $r^{[k_3-\ell_{\mathcal{T}},k_3]} = r^{[k_4-\ell_{\mathcal{T}},k_4]}$ after we have reached the last window.

For temporal concepts of the form $\bigcirc C$ and $\bigcirc^- C$, $C\,\mathcal{U}_{[c_1,c_2]}D$ and $C\,\mathcal{S}_{[c_1,c_2]}D$, our window size is sufficient for determining if Conditions **R1**–**R4** are satisfied. For concepts of the form $C\,\mathcal{S}_{[c_1,\infty)}D$, we just verify that $C\,\mathcal{S}_{[0,\infty)}D$ is present c_1 types before, and that every $C\,\mathcal{S}_{[0,\infty)}D$ is preceded by either D or $C\,\mathcal{S}_{[0,\infty)}D$. (Whether these $\mathcal{S}$-expressions are eventually realised by an occurrence of the concept D in the past is checked when we verify the run in the other direction.) Concepts of the form $C\,\mathcal{U}_{[c,\infty)}D$ are kept in memory until they are realised. In addition, we verify that every concept of the form $C\,\mathcal{U}_{[c,\infty)}D \in r(k_3)$ is realised before we reach $r(k_4)$, thus ensuring that any of these concepts in $r(k_4)$ can be realised in the repetition of the sequence $\big(r(k_3+1)\dots r(k_4)\big)^{\omega}$. Note that, in each step, we only need to store an exponential amount of information. Therefore, this approach can be implemented by a non-deterministic exponentially-space bounded Turing machine. In a similar manner, but treating $\mathcal{S}$ as $\mathcal{U}$ and vice versa, we can show that, starting from the initial window $r^{[0,k_3]}$, we can find a regular sequence into the past of the form $^{\omega}\big(r(-k_1)\dots r(-k_2-1)\big)\, r(-k_2)\dots r(0)\dots r(k_3)$ so that the whole procedure runs in exponential space.

If $\mathcal{T}$ contains only $\mathrm{LTL}_{\mathcal{ALC}}$ concepts, then, by Lemma A.3, k_1, k_2, k_3 and k_4 can be bounded *single* exponentially in the size of $\Box\Box^-\mathcal{T}$. Furthermore, $\ell_{\mathcal{T}}$ is 1, and thus instead of sequences of types we need to keep in memory only three *single* types (the current type, the next type, and the first repeating types $r(k_3+1)$ and $r(-k_2-1)$), each of which is of polynomial size. Finally, checking that the types we guess belong to the exponentially large quasistate Q can be done in exponential time, and hence the whole procedure runs in exponential time. □

Lemma A.5. *Satisfiability is ExpSpace-complete in* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$. *Moreover, it is ExpTime-complete in* $\mathrm{LTL}_{\mathcal{ALC}|gGCI}$.

Proof. The first lower bound follows from the fact that satisfiability of propositional $\mathrm{LTL}^{\mathsf{bin}}$ formulae over the integers is already ExpSpace-complete [29], while the second lower bound follows from the complexity of $\mathcal{ALC}$ TBox satisfiability [43]. The upper bounds are established by the following procedure. Let $\Box\Box^-\mathcal{T}$ be an $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}|gGCI}$ formula. Since we only need to verify the existence of a constant quasimodel, it suffices to construct a single quasistate, which we do by type elimination. For this, we start with the set of all possible concept types, from which we step-by-step remove those that do not satisfy the conditions for the quasistate in a constant quasimodel. It is easy to verify that, for each concept type, this can be decided deterministically, and that for all but Condition **M3**, we only have to check the remaining other concept types in the current set. For

Condition **M3**, we have to check that for every concept type t in the current quasistate Q, there is a run over Q that goes through t. By Lemma A.4, this can be verified in exponential space, and in exponential time provided that $\mathcal{T}$ contains only $\mathrm{LTL}_{\mathcal{ALC}}$ concepts. We thus obtain the desired complexity bounds. □

B PROOFS FOR SECTION 4

Lemma 4.2. *In the presence of interval-rigid names, satisfiability of* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae (with global GCIs) can be polynomially reduced to satisfiability of* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formulae (with global GCIs and) without assertions.*

Proof. We can use the same constructions as for Theorems 3.1 and 3.2, provided that we make the new concept names $A_{B(a)}$ interval-rigid whenever $B \in \mathsf{N_C}$ is (with $\mathsf{iRig}(A_{B(a)}) := \mathsf{iRig}(B)$), and similarly for $A_{r(a,b)}$, r_a and $r \in \mathsf{N_R}$ (with $\mathsf{iRig}(A_{r(a,b)}) = \mathsf{iRig}(r_a) = \mathsf{iRig}(r)$). All interpretations constructed in the respective proofs then respect all interval-rigid names. □

To prove Lemma 4.3, we first show that the existence of quasimodels completely captures satisfiability of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formulae.

Lemma B.1. *An* $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ *formula* φ *is satisfiable with interval-rigid names iff there is a quasimodel for* φ.

Proof. ($\Rightarrow$) Assume there is a model $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ of φ that respects interval-rigid names. We associate a concept run r_d to every domain element $d \in \Delta^{\mathfrak{I}}$ by setting

$$r_d(i) := \{C \in \mathsf{cl^c}(\varphi) \mid d \in C^{\mathcal{I}_i}\}.$$

We define the formula run $r_{\mathfrak{I}}$ by $r_{\mathfrak{I}}(i) := \{\psi \in \mathsf{cl^f}(\varphi) \mid \mathfrak{I}, i \models \psi\}$ for all $i \in \mathbb{Z}$. We now set $\mathfrak{R} := \{r_d \mid d \in \Delta^{\mathfrak{I}}\} \cup \{r_{\mathfrak{I}}\}$, and define the infinite sequence of quasistates $S(i)$, $i \in \mathbb{Z}$, as follows. For $i \in \mathbb{Z}$, the set $\mathcal{R}_{S(i)}$ contains all run segments $r^{[i-\ell_\varphi, i+\ell_\varphi]}$ with $r \in \mathfrak{R}$. Furthermore, $\mathcal{C}_{S(i)}$ is the set of role constraints $\sigma \overset{s}{_k} \sigma'$, such that $\sigma = r_d^{[i-\ell_\varphi, i+\ell_\varphi]}$, $\sigma' = r_{d'}^{[i-\ell_\varphi, i+\ell_\varphi]}$ and $k \in [1, \mathsf{iRig}(s)]$. It is straightforward to show that $(S, \mathfrak{R})$ is a quasimodel. In particular, the maximal compatibility relations π_i, $i \in \mathbb{Z}$, contain the pairs $(r^{[i-\ell_\varphi, i+\ell_\varphi]}, r^{[i+1-\ell_\varphi, i+1+\ell_\varphi]})$ with $r \in \mathfrak{R}$.

($\Leftarrow$) Assume there is a quasimodel $(S, \mathfrak{R})$ for φ. We define the temporal DL interpretation $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_i)_{i\in\mathbb{Z}})$ as follows.

$$\begin{aligned}
\Delta^{\mathfrak{I}} &:= \{d_r \mid r \in \mathfrak{R} \text{ is a concept run}\}\\
A^{\mathcal{I}_i} &:= \{d_r \mid A \in r(i),\ r \in \mathfrak{R}\}\\
s^{\mathcal{I}_i} &:= \{(d_r, d_{r'}) \mid r^{[j-\ell_\varphi, j+\ell_\varphi]} \overset{s}{_1} r'^{[j-\ell_\varphi, j+\ell_\varphi]} \in S(j),\ i-j \in [0, \mathsf{iRig}(s)),\ r, r' \in \mathfrak{R}\}
\end{aligned}$$

By referring to the role constraint $r^{[j-\ell_\varphi, j+\ell_\varphi]} \overset{s}{_1} r'^{[j-\ell_\varphi, j+\ell_\varphi]}$, which may lie in the past of time point i, we make sure that we always connect the domain elements for the same two runs for $\mathsf{iRig}(s)$ consecutive time points, and do not use a different pair of runs for every time point. To see that $\mathfrak{I}$ respects interval-rigid names, consider any $(d_r, d_{r'}) \in s^{\mathcal{I}_i}$. There must be a j with $i - j \in [0, \mathsf{iRig}(s))$ and $r^{[j-\ell_\varphi, j+\ell_\varphi]} \overset{s}{_1} r'^{[j-\ell_\varphi, j+\ell_\varphi]} \in S(j)$. But then $(d_r, d_{r'}) \in s^{\mathcal{I}_\ell}$ for all $\ell \in [j, j + \mathsf{iRig}(s))$, and this interval includes i.

To show that $\mathfrak{I}$ is also a model of φ, we prove the following claim.

Claim. *For all concept runs* $r \in \mathfrak{R}$, $C \in \mathsf{cl^c}(\varphi)$ *and* $i \in \mathbb{Z}$, *we have*

$$C \in r(i) \quad \text{iff} \quad d_r \in C^{\mathcal{I}_i}.$$

Proof of the claim. We argue by structural induction. If C is a concept name, then the claim holds by construction. Moreover, the cases $C = \neg D$ and $C = D \sqcap E$ are straightforward. It remains to consider $\exists$, $\bigcirc$, $\bigcirc^-$, $\mathcal{U}_I$, and $\mathcal{S}_I$.

- Assume $C = \exists s.D$: if $\exists s.D \in r(i)$, then, by **M2** and **S2**, there is a run segment $\sigma \in \mathcal{R}_{S(i)}$ such that $r^{[i-\ell_\varphi, i+\ell_\varphi]} \overset{s}{k} \sigma \in \mathcal{C}_{S(i)}$, $D \in \sigma(0)$, and $k \in [1, \mathrm{iRig}(s)]$. By repeated application of **M2**, **C3**, and **C5**, we can find another run segment $\sigma' \in \mathcal{R}_{S(i-(k-1))}$ such that $r^{[i-(k-1)-\ell_\varphi, i-(k-1)+\ell_\varphi]} \overset{s}{1} \sigma' \in \mathcal{C}_{S(i-(k-1))}$ and $\sigma'^{\geq k-1} = \sigma^{\leq \ell_\varphi - (k-1)}$. By **M3**, there must be a run $r' \in \mathfrak{R}$ such that $r'^{[i-(k-1)-\ell_\varphi, i-(k-1)+\ell_\varphi]} = \sigma'$ and $D \in r'(i)$. By the induction hypothesis and the definition of $s^{\mathcal{I}_i}$, we obtain $d_{r'} \in D^{\mathcal{I}_i}$ and $(d_r, d_{r'}) \in s^{\mathcal{I}_i}$, and hence $d_r \in (\exists s.D)^{\mathcal{I}_i}$. Conversely, if $d_r \in (\exists s.D)^{\mathcal{I}_i}$, then there is $r' \in \mathfrak{R}$ such that $(d_r, d_{r'}) \in s^{\mathcal{I}_i}$ and $d_{r'} \in D^{\mathcal{I}_i}$. By the induction hypothesis and the definition of $s^{\mathcal{I}_i}$, we have $D \in r'(i)$ and furthermore also $r^{[j-\ell_\varphi, j+\ell_\varphi]} \overset{s}{1} r'^{[j-\ell_\varphi, j+\ell_\varphi]} \in \mathcal{C}_{S(j)}$ for some j with $i - j \in [0, \mathrm{iRig}(s))$. By repeated application of **C3** and **C4**, we obtain a role constraint $r^{[i-\ell_\varphi, i+\ell_\varphi]} \overset{s}{i-j+1} \sigma'$, where $\sigma' \in \mathcal{R}_{S(i)}$ is such that $\sigma'^{\leq j+\ell_\varphi - i} = r'^{[i-\ell_\varphi, j+\ell_\varphi]}$. Hence, we have $D \in r'(i) = \sigma'(0)$, and thus $\exists s.D \in r(i)$ by **C1**.
- Assume $C = \bigcirc D$: we have that $\bigcirc D \in r(i)$ iff $D \in r(i+1)$ (by **R1**) iff $d_r \in D^{\mathcal{I}_{i+1}}$ (by induction) iff $d_r \in (\bigcirc D)^{\mathcal{I}_i}$.
- Assume $C = \bigcirc^- D$: we have that $\bigcirc^- D \in r(i)$ iff $D \in r(i-1)$ (by **R2**) iff $d_r \in D^{\mathcal{I}_{i-1}}$ (by induction) iff $d_r \in (\bigcirc^- D)^{\mathcal{I}_i}$.
- Assume $C = D \mathcal{U}_I E$: we have that $D \mathcal{U}_I E \in r(i)$ iff there is j such that $j - i \in I$, $E \in r(j)$ and $D \in r(\ell)$ for all $\ell \in [i, j)$ (by **R3** and **R5**); by induction, this happens iff $d_r \in E^{\mathcal{I}_j}$ and $d_r \in D^{\mathcal{I}_i}$ for all $\ell \in [i, j)$, which is equivalent to $d_r \in (D \mathcal{U}_I E)^{\mathcal{I}_i}$.
- Assume $C = D \mathcal{S}_I E$: we have that $D \mathcal{S}_I E \in r(i)$ iff there is j such that $i - j \in I$, $E \in r(j)$ and $D \in r(\ell)$ for all $\ell \in (j, i]$ (by **R4** and **R6**); by induction, this happens iff $d_r \in E^{\mathcal{I}_j}$ and $d_r \in D^{\mathcal{I}_i}$ for all $\ell \in (j, i]$, which is equivalent to $d_r \in (D \mathcal{S}_I E)^{\mathcal{I}_i}$.

Using **S3** and similar arguments as above, we can now show that $\mathfrak{I}, i \models \psi$ iff $\psi \in r_{\mathfrak{I}}(i)$ for all $\psi \in \mathrm{cl}^{\mathrm{f}}(\varphi)$, where $r_{\mathfrak{I}}$ is the unique formula run in $\mathfrak{R}$. Hence, by **M1**, we obtain that $\mathfrak{I}, 0 \models \varphi$. □

This concludes the proof of Lemma B.1. □

Before we show Lemma 4.3, we prove the following two auxiliary results.

Lemma B.2. *Let $(S, \mathfrak{R})$ be a quasimodel for φ such that $S(n) = S(m)$ for $n < m$. Then $(S', \mathfrak{R}')$ with $S' := S^{\leq n} \cdot S^{>m}$ and*

$$\mathfrak{R}' := \{r_1^{\leq n} \cdot r_2^{>m} \mid r_1, r_2 \in \mathfrak{R},\ r_1^{[n-\ell_\varphi, n+\ell_\varphi]} = r_2^{[m-\ell_\varphi, m+\ell_\varphi]}\}$$

is also a quasimodel for φ.

Proof. We show that $(S', \mathfrak{R}')$ satisfies Conditions **M1**–**M3**. First note that, since $(S, \mathfrak{R})$ is a quasimodel for φ, $(S', \mathfrak{R}')$ satisfies **M1**. Also, for any $r_1, r_2 \in \mathfrak{R}$ with $r_1^{[n-\ell_\varphi, n+\ell_\varphi]} = r_2^{[m-\ell_\varphi, m+\ell_\varphi]}$, $r_1^{\leq n} \cdot r_2^{>m}$ is also a run. Consequently, $\mathfrak{R}'$ is a set of runs. Since $S(n) = S(m)$, for every $r_1 \in \mathfrak{R}$ there is a run $r_2 \in \mathfrak{R}$ that satisfies $r_1^{[n-\ell_\varphi, n+\ell_\varphi]} = r_2^{[m-\ell_\varphi, m+\ell_\varphi]}$ and vice versa (by swapping n and m). It follows that $(S', \mathfrak{R}')$ satisfies **M3**. Finally, as $S(n) = S(m)$, the pair $(S(n), S(m+1))$ is compatible, and since we require $r_1^{[n-\ell_\varphi, n+\ell_\varphi]} = r_2^{[m-\ell_\varphi, m+\ell_\varphi]}$, we have **M2** for $(S', \mathfrak{R}')$. □

In the following, let $\sharp_\varphi$ denote the size of the (double exponential) set of all possible run segments and role constraints for φ.

Lemma B.3. *Formula φ has a quasimodel iff there is a sequence of quasistates*

$$S(-k_1) \dots S(-k_2 - 1)\, S(-k_2) \dots S(0) \dots S(k_3)\, S(k_3 + 1) \dots S(k_4)$$

such that

- $k_1, k_2, k_3, k_4 \leq (|\varphi| \cdot \sharp_\varphi^2 + 1) \cdot 2^{\sharp_\varphi}$;
- *for all* $i \in [-k_1, k_4)$, *the pair* $(S(i), S(i+1))$ *is compatible, with* π_i *being the maximal compatibility relation witnessing this (cf.* **C2–C5***);*
- $S(-k_1) = S(-k_2)$ *and* $S(k_3) = S(k_4)$*;*
- $S(0)$ *satisfies* **M1***;*
- *for every* $\sigma_1 \in \mathcal{R}_{S(k_3)}$, *there is a sequence* $r = \sigma_1(0)\,\sigma_2(0) \dots \sigma_{k_4-k_3}(0)$ *that realises all* $\mathcal{U}$*-expressions in* $\sigma_1(0)$ *such that* $(\sigma_i, \sigma_{i+1}) \in \pi_{k_3+i}$ *for all* $i \in [0, k_4 - k_3)$*;*
- *for every* $\sigma_1 \in \mathcal{R}_{S(-k_2)}$, *there is a sequence* $r = \sigma_{k_1-k_2}(0) \dots \sigma_2(0)\,\sigma_1(0)$ *that realises all* $\mathcal{S}$*-expressions in* $\sigma_1(0)$ *such that* $(\sigma_{i+1}, \sigma_i) \in \pi_{-k_2-i}$ *for all* $i \in [0, k_1 - k_2)$.

Proof. ($\Rightarrow$) Let $(S, \mathfrak{R})$ be a quasimodel for φ. Observe that the number of possible quasistates is bounded by $2^{\sharp_\varphi}$. We can assume w.l.o.g. that there are some $k_2, k_3 \leq 2^{\sharp_\varphi}$ such that all quasistates in $S^{\leq k_2}$ and $S^{\geq k_3}$ occur infinitely often. If this is not the case, we can simply take k_2 as the minimal number such that $S(m) \neq S(k_2)$ holds for all $m < k_2$, and use Lemma B.2 to remove repeating quasistates in $S^{[-k_2,0]}$. For the right side of the sequence, we can similarly take k_3 as the maximal number such that $S(m) \neq S(k_3)$ holds for all $m > k_3$, and use Lemma B.2 to remove repeating quasistates in $S^{[0,k_3]}$.

Consider now an arbitrary $\alpha\,\mathcal{U}_I\beta \in \sigma(0)$ for some $\sigma \in \mathcal{R}_{S(k_3)}$, and take any $r \in \mathfrak{R}$ with $r^{[k_3-\ell_\varphi, k_3+\ell_\varphi]} = \sigma$, which must exist by **M3**. Let $m' \geq k_3$ be the minimal number such that $r^{[k_3,k_3+m']}$ realises $\alpha\,\mathcal{U}_I\beta$. Assume now that there are i, j such that $k_3 < i < j < m'$, $r^{[i-\ell_\varphi, i+\ell_\varphi]} = r^{[j-\ell_\varphi, j+\ell_\varphi]}$, and $S(i) = S(j)$. By Lemma B.2, there is a quasimodel $(S', \mathfrak{R}')$ for φ such that $S' = S^{\leq i} \cdot S^{>j}$ and $r^{\leq i} \cdot r^{>j}$ is a run in $\mathfrak{R}'$. It follows that we can construct a quasimodel $(S_1, \mathfrak{R}_1)$ for φ with $S_1^{\leq k_3} = S^{\leq k_3}$ and a run $r_1 \in \mathfrak{R}_1$ such that $r_1^{[k_3-\ell_\varphi, k_3+\ell_\varphi]} = \sigma \in \mathcal{R}_{S_1(k_3)} = \mathcal{R}_{S(k_3)}$ and $\alpha\,\mathcal{U}_I\beta$ is realised by the subsequence $r_1^{[k_3,k_3+m_1]}$, for some $m_1 \leq \sharp_\varphi \cdot 2^{\sharp_\varphi}$ ($2^{\sharp_\varphi}$ is the number of possibilities for $S(i)$, and $\sharp_\varphi$ is the number of possibilities for $r^{[i,i+\ell_\varphi]}$).

Then, we consider the next expression of the form $\alpha'\,\mathcal{U}_{I'}\beta' \in \sigma(0)$ and assume that $r_1^{[k_3,k_3+m'']}$ realises it for some minimal $m'' \geq m_1$. Using the same construction as above, we obtain a quasimodel $(S_2, \mathfrak{R}_2)$ for φ with $S_2^{\leq k_3+m_1} = S_1^{\leq k_3+m_1}$ and a run $r_2 \in \mathfrak{R}_2$ such that $r_2^{[k_3-\ell_\varphi, k_3+\ell_\varphi]} = \sigma \in \mathcal{R}_{S(k_3)}$ and $r_2^{[k_3,k_3+m_2]}$ realises both $\alpha\,\mathcal{U}_I\beta$ and $\alpha'\,\mathcal{U}_{I'}\beta'$, for some $m_2 \leq 2 \cdot \sharp_\varphi \cdot 2^{\sharp_\varphi}$. We can proceed in this way and construct a quasimodel containing a run through σ that realises all $\mathcal{U}$-expressions in $\sigma(0)$ after at most $|\varphi| \cdot \sharp_\varphi \cdot 2^{\sharp_\varphi}$ steps.

After that, we consider in the same manner another run segment $\sigma' \in \mathcal{R}_{S(k_3)}$. To realise all $\mathcal{U}$-expressions in some run through σ', we need at most $|\varphi| \cdot \sharp_\varphi \cdot 2^{\sharp_\varphi}$ additional steps. Since there are at most $\sharp_\varphi$ run segments in $\mathcal{R}_{S(k_3)}$, we require at most $|\varphi| \cdot \sharp_\varphi^2 \cdot 2^{\sharp_\varphi}$ steps to realise all until-expressions in all run segments in $\mathcal{R}_{S(k_3)}$. We now take another $2^{\sharp_\varphi}$ steps to find a time point $k_4 \leq (|\varphi| \cdot \sharp_\varphi^2 + 1) \cdot 2^{\sharp_\varphi}$ such that $S^*(k_3) = S^*(k_4)$ in the resulting quasimodel $(S^*, \mathfrak{R}^*)$ (since we assumed that $S(k_3)$ occurs infinitely often). By our construction, all $\mathcal{U}$-expressions in $S^*(k_3)$ can be realised by runs through $S^{*[k_3,k_4]}$. With a similar argument we can modify the quasimodel $(S^*, \mathfrak{R}^*)$ so that all $\mathcal{S}$-expressions in $S^*(-k_2)$ realised by runs through $S^{*[-k_1,-k_2]}$, for some $k_1 \leq (|\varphi| \cdot \sharp_\varphi^2 + 1) \cdot 2^{\sharp_\varphi}$. Hence, the sequence $S^{*[-k_1,k_4]}$ satisfies all conditions required by the lemma.

($\Leftarrow$) Let now $S^*(-k_1) \dots S^*(-k_2 - 1)\,S^*(-k_2) \dots S^*(0) \dots S^*(k_3)\,S^*(k_3 + 1) \dots S^*(k_4)$ be a sequence with the given properties. We construct a quasimodel $(S, \mathfrak{R})$ for φ, where S is defined by

$$S = {}^{\omega}\big(S^*(-k_1) \dots S^*(-k_2 - 1)\big)\, S^*(-k_2) \dots S^*(0) \dots S^*(k_3)\, \big(S^*(k_3 + 1) \dots S^*(k_4)\big)^{\omega},$$

and $\mathfrak{R}$ contains all sequences of types of the form

$$\begin{aligned}
&\dots \cdot \sigma^{-2}_{-k_1}(0) \dots \sigma^{-2}_{-k_2-1}(0) \cdot \\
&\quad \sigma^{-1}_{-k_1}(0) \dots \sigma^{-1}_{-k_2-1}(0) \cdot \\
&\quad \sigma^{0}_{-k_1}(0) \dots \sigma^{0}_{-k_2-1}(0) \cdot \qquad \sigma^{0}_{-k_2}(0) \dots \sigma^{0}_{0}(0) \dots \sigma^{0}_{k_3}(0) \qquad \cdot \sigma^{0}_{k_3+1}(0) \dots \sigma^{0}_{k_4}(0) \cdot \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \sigma^{1}_{k_3+1}(0) \dots \sigma^{1}_{k_4}(0) \cdot \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \sigma^{2}_{k_3+1}(0) \dots \sigma^{2}_{k_4}(0) \cdot \ \dots,
\end{aligned}$$

where each σ^j_i is an element of $S^*(i)$, each pair of adjacent run segments in this sequence is contained in the corresponding compatibility relation (for $\sigma^j_{k_4}$ and $\sigma^{j+1}_{k_3+1}$ we consider π_{k_3}, and for $\sigma^{j-1}_{-k_2-1}$ and $\sigma^j_{-k_1}$ we consider π_{-k_2}), and there are infinitely many $j \geq 1$ for which $\sigma^{j-1}_{k_4}(0)\, \sigma^j_{k_3+1}(0) \dots \sigma^j_{k_4}(0)$ realises all $\mathcal{U}$-expressions in $\sigma^{j-1}_{k_4}(0)$, and infinitely many $-j \leq -1$ for which all $\mathcal{S}$-expressions in $\sigma^{-j+1}_{-k_1}(0)$ are realised by $\sigma^{-j}_{-k_1}(0) \dots \sigma^{-j}_{-k_2-1}(0)\, \sigma^{-j+1}_{-k_1}(0)$.

Due to **C3**, for all $r \in \mathfrak{R}$, each subsequence of length $\ell_\varphi + 1$ is a run segment from one of the sets $S^*(i)$. To show that r is a run, we verify Condition **R5**. But this follows from the local Condition **R3**, which states that each $\mathcal{U}$-expression is either satisfied by the next ℓ_φ types in r, or it is deferred; an analogous argument holds for $\mathcal{S}$-expressions. If they are deferred, then our construction ensures that $\mathcal{U}$- and $\mathcal{S}$-expressions are not deferred indefinitely.

It remains to show that $(S, \mathfrak{R})$ satisfies Conditions **M1**–**M2**. Condition **M2** is immediately satisfied by our construction of $\mathfrak{R}$. Condition **M1** only concerns $S(0)$, and is also satisfied since $S(0) = S^*(0)$ by construction. For Condition **M3**, for every $i \in \mathbb{Z}$ and $\sigma \in S(i)$, we need to find a run $r \in \mathfrak{R}$ with $r^{[i-\ell_\varphi, i+\ell_\varphi]} = \sigma$. By **C2** and **C3**, we can extend σ to the left and to the right. When we extend to the right direction, we need to make sure that, whenever we reach a $\sigma' \in S^*(k_4)$, we continue it in such a way that all $\mathcal{U}$-expressions in σ' are realised by the next $k_4 - k_3$ types; and similarly for the left direction, where we check whether the $\mathcal{S}$-expressions are realised. Note that the resulting run is contained in our set $\mathfrak{R}$ of runs by construction. This is always possible by our assumptions on the sequence $S^*(-k_1) \dots S^*(-k_2 - 1)\, S^*(-k_2) \dots S^*(0) \dots S^*(k_3)\, S^*(k_3 + 1) \dots S^*(k_4)$. □

One side result of the constructions in the above proof is that, for a satisfiable formula, we can always find a regular quasimodel.

Corollary B.4. *If φ has a quasimodel, then it has a quasimodel $(S, \mathfrak{R})$ in which S is of the form*

$$^{\omega}\big(S(-k_1) \dots S(-k_2 - 1)\big)\, S(-k_2) \dots S(0) \dots S(k_3)\, \big(S(k_3 + 1) \dots S(k_4)\big)^{\omega},$$

such that k_1, k_2, k_3 and k_4 are bounded triple exponentially in the size of φ and $\mathsf{iRig}|_\varphi$.

Lemma 4.3 directly follows from Lemma B.1 and Corollary B.4. Based on this result, we can show 2-ExpSpace-completeness of satisfiability of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$-formulae.

Theorem 4.4. *Satisfiability in $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ with interval-rigid names is 2-ExpSpace-complete.*

Proof. Since the lower bound follows from a known result [32] (with an adaptation to $\mathbb{Z}$), we only need to prove the upper bound. By Lemmas B.1 and B.3, it suffices to decide the existence of a sequence of quasistates with certain properties. We show how the latter can be decided non-deterministically in double exponential space by guessing such a sequence, quasistate after quasistate. The basic observation is that storing a single quasistate requires at most double exponential space, and that we only require a constant number of quasistates in memory at any point of the computation.

Since we deal with the timeline $\mathbb{Z}$, we need to check the regularity conditions in both directions. We explain how to check the conditions for the right side of the quasimodel. The conditions for the left side are analogous. We first guess the four numbers k_1, k_2, k_3 and k_4 from Lemma B.3, which can be stored using double exponentially many bits. We then guess and store the quasistate $S(k_3)$, and for each $\sigma \in \mathcal{R}_{S(k_3)}$ we keep a list of $\mathcal{U}$-expressions that need to be satisfied, which initially contains all $\alpha\,\mathcal{U}\beta \in \sigma(0)$.

We now guess the first k_3 quasistates $S(i)$ one after another, so that Condition **M1** is satisfied for $S(0)$, and each pair $(S(i), S(i+1))$ of consecutive quasistates is compatible. For this, we only need to keep two consecutive quasistates in memory at any point. We then guess the next $k_4 - k_3$ quasistates as before, where we additionally test whether there are runs satisfying all required $\mathcal{U}$-expressions. For this, it suffices to guess which run segment of the next quasistate corresponds to a run starting with σ; this, of course, has to be compatible with the run segment guessed for the previous quasistate. We remove any $\mathcal{U}$-expression from our list that is satisfied by this guessed run segment, and require that all $\mathcal{U}$-expressions must have been removed when we reach $S(k_4)$. This can be done in double exponential space. Finally, after we have guessed $S(k_4)$, we verify that $S(k_4) = S(k_3)$. The procedure to check the conditions on the left side is similar, but we check whether the $\mathcal{S}$-expressions are satisfied when we go from the quasistate $S(-k_2)$ to the quasistate $S(-k_1)$. By Lemmas B.1 and B.3, our algorithm is sound and complete. Since at each step, we only require double exponential space, we thus establish that satisfiability of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ formulae with interval-rigid names is in 2-ExpSpace. □

Theorem 4.7. *Satisfiability in* $\mathrm{LTL}_{\mathcal{ALC}|gGCI}$ *with interval-rigid names is 2-ExpTime-hard.*

Proof. We consider the reduction used for the sublogic $\mathcal{ALC}$-LTL of $\mathrm{LTL}_{\mathcal{ALC}}$ [20, Lemma 4.2], and observe that the rigid concept and role names used in that reduction need to stay rigid only from time point 0 to time point $2^k - 1$, where k is polynomial in the size of the original problem. Since 2^k can be written using polynomially many bits, we can designate these names to be 2^k-rigid. However, we also need to do this for all *negations* of the rigid concept names A to ensure that if A is implied to hold at a domain element d at some point in the interval $[0, 2^k - 1]$, then this information is also propagated backwards in time, i.e., d satisfies A also at time point 0. One can make $\neg A$ 2^k-rigid by introducing a fresh 2^k-rigid concept name A' and adding the global GCI $A' \equiv \neg A$ to the formula. This is not needed for the single rigid role r used in the proof, since all necessary r-connections are already implied at time point 0.

The main structure used in the proof is a tree whose root is described by the individual name a and whose edges are described by the rigid role r. This role connects all domain elements relevant for the reduction. All these r-connections are already declared at time point 0, but to ensure that they are present throughout the whole time interval $[0, 2^k - 1]$, we further need to encode that none of these r-connections can exist at time point -1. Since we are only interested in the domain elements reachable via r from a, we can use r to propagate a fresh concept name B to all relevant domain elements via the assertion $B(a)$ and the global GCI $B \sqsubseteq \forall r.B$, and then use the global GCI $\bigcirc B \sqsubseteq \neg\exists r.\top$ to enforce that these r-connections do not exist at time point -1, and hence must exist in the whole interval $[0, 2^k - 1]$ since r is 2^k-rigid. Similarly, we can express that the value of a 2^k-rigid concept name A (where $\neg A$ is also 2^k-rigid) must stay constant in $[0, 2^k - 1]$ via the global GCIs $\bigcirc B \sqsubseteq (\neg A) \leftrightarrow (\bigcirc A)$. □

C PROOFS FOR SECTION 5

Theorem 5.4. *Satisfiability in $\mathcal{ALC}$-LTL with interval-rigid names is 2-ExpSpace-hard.*

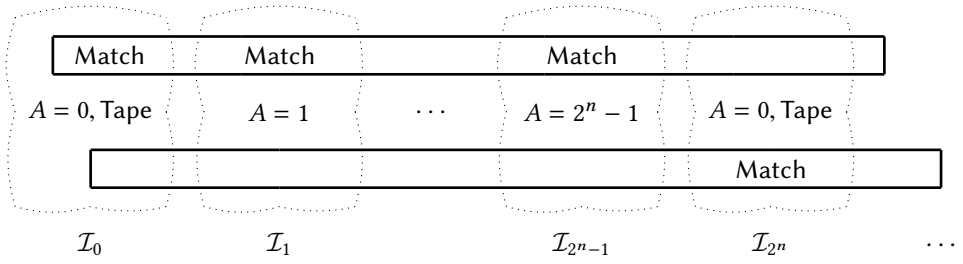


Fig. 3. A model of the $\mathcal{ALC}$-LTL formula $\varphi_{\mathcal{M},w}$ (for simplicity, some symbols have been omitted).

Proof. The proof is by reduction from the word problem for double exponentially space bounded DTMs. Let $\mathcal{M} = (Q, \Sigma, \Gamma, \Theta, q_0, F)$ be a DTM, where

- Q is a finite set of states,
- Σ is the input alphabet,
- $\Gamma \supseteq \Sigma$ is the work alphabet containing the blank symbol $b \notin \Sigma$,
- $\Theta\colon Q \times \Gamma \to Q \times \Gamma \times \{l, r\}$ is the transition function,
- q_0 is the initial state, and
- $F \subseteq Q$ is the set of accepting states.

As usual, a configuration of $\mathcal{M}$ is a word wqw' with $w, w' \in \Gamma^*$ and $q \in Q$, meaning that the tape contains the word ww', the machine is in state q and the head is on the position of the left-most symbol of w'. To reflect the space bounds of $\mathcal{M}$, we assume that all configurations wqw' satisfy $|ww'| \leq 2^{2^n}$, where $n = p(m)$ for inputs of length m and some fixed polynomial p. We further assume w.l.o.g. that $\mathcal{M}$ does not attempt to move to the left (right) when the head is on the left-most (right-most) tape position.

We construct an $\mathcal{ALC}$-LTL$^{\bigcirc}$ formula $\varphi_{\mathcal{M},w}$ that is satisfiable with interval-rigid names iff $\mathcal{M}$ accepts w. Using Lemma 5.1, we can then transform $\varphi_{\mathcal{M},w}$ into an equisatisfiable $\mathcal{ALC}$-LTL formula of polynomial size. For convenience, we also use assertions using the $\mathcal{U}$-operator, e.g., $(A \sqcap (B\,\mathcal{U}\,C))(a)$, which can equivalently be expressed using the formula $A(a) \wedge (B(a)\,\mathcal{U}\,C(a))$. In all cases where we use such expressions, it is easily verified that they are equivalent to some $\mathcal{ALC}$-LTL formulae.

The general idea of the construction is as follows. We encode the sequence of configurations along the timeline, where each configuration is represented by $2^n \cdot 2^{2^n}$ successive time points. Each tape cell is represented by 2^n successive time points, in which the double exponential tape address is stored and the first time point is marked with the concept Tape. To represent the tape address, we use a binary counter to mark each time point with a *bit position*, and use a concept name C_{Bit} to store the bit value of the tape address at this bit position. Using a $2^n + 1$-rigid role r and another binary counter, we can link each bit position with the corresponding bit position for the next tape cell, which is used to implement the double exponential counter for the tape addresses.

To transfer information between successive configurations, we additionally represent all different tape addresses using a set of 2^{2^n} designated domain elements which we call *matcher objects*. Each matcher object stores its associated tape address using 2^n r-successors for the different bit positions. A concept Match is used to identify which matcher object is associated with the current tape address. This way, we can use the matcher objects to transfer the information of each tape cell to the next configuration (see Figure 3).

In the reduction, we use the following symbols:

- an individual name a for a domain element associated with the double exponential counter,
- a concept name Tape to mark tape cells,

- a concept name Init to mark the beginning of a configuration,
- a concept name Head to mark the head position of a configuration,
- concept names C_σ and N_σ, with $\sigma \in \Gamma$, to represent the symbol at the *c*urrent tape cell and the *n*ext tape cell to the right, respectively,
- concept names C_q and N_q, with $q \in Q' := Q \cup \{\overline{q}\}$, to represent the states associated with the *c*urrent tape cell and the *n*ext tape cell to the right, respectively, where $\overline{q}$ expresses that the head is at another tape position,
- concept names $A_0, \dots, A_{n-1}$ to encode the binary representation of the bit positions of the double exponential values of the tape addresses, where A_0 encodes the least significant bit,
- the bit values of the *c*urrent double exponential tape address are represented using a concept name C_{Bit},
- a $2^n + 1$-rigid concept name N_{Bit} is used to encode the value of C_{Bit} in the corresponding bit position of the *n*ext tape address,
- a concept name C'_{Bit}, which we make rigid using the expressivity of $\mathcal{ALC}$-LTL$^{\bigcirc}$, to store the bit values for the matcher objects,
- a $2^n + 1$-rigid role r and concept names $B_0, \dots, B_n$ to transfer the truth values of N_{Bit} and C'_{Bit} exponentially far away (the $2^n + 1$-rigid role ensures that we can reach the *same* bit position exponentially far way and encode a counter with exponentially many bits),
- a concept name Match, to mark matcher objects representing the current tape address, and
- concept names C'_σ, N'_σ and C'_q, N'_q to mark the matcher objects with information about the *c*urrent state and *n*ext state for the associated tape cell.

We are now ready to start our construction. The formula $\varphi_{\mathcal{M},w}$ is defined as the conjunction of a set of formulae which we describe one after the other.

We first describe how we implement the double exponential counter for the tape addresses. We denote by $(A = k)$ the polynomial-sized concept that identifies the value of the A-counter with k, i.e., that refers to the k-th bit position of the double exponential counter. We implement the behaviour of Tape and $A_0, \dots, A_{n-1}$ with

$$(\top \sqsubseteq \mathsf{Tape}) \wedge \Box\big(\mathsf{Tape} \equiv (A = 0)\big), \tag{1}$$

$$\bigwedge_{i<n} \Box\Big(\prod_{j<i} A_j \sqsubseteq (A_i \leftrightarrow \bigcirc\neg A_i)\Big), \tag{2}$$

$$\bigwedge_{i<n} \Box\Big(\bigsqcup_{j<i} \neg A_j \sqsubseteq (A_i \leftrightarrow \bigcirc A_i)\Big). \tag{3}$$

These axioms express that

- Tape is satisfied by all domain elements at time point 0, which also starts the A-counter at all domain elements in a synchronised fashion;
- the A-counter is incremented in each time step, which also happens synchronously at all domain elements (cf. the concepts $(A^F{-}{-})$ in the proof of Theorem 3.4); and
- Tape is satisfied (by all domain elements) exactly after each 2^n time points, namely every time the A-counter overflows from $2^n - 1$ to 0.

Moreover, each time point stores the bit value of the bit position given by the A-counter, using the concept name C_{Bit}. This information is shared by all domain elements:

$$\Box\big((\top \sqsubseteq C_{\mathsf{Bit}}) \vee (C_{\mathsf{Bit}} \sqsubseteq \bot)\big).$$

To transfer the value of C_{Bit} to the next time point where the current value of the A-counter reoccurs, we use the $2^n + 1$-rigid role r and concept names $B_0, \dots, B_n$ to implement a fresh counter for each bit position. To increment the B-counter, we include in our formulation of $\varphi_{\mathcal{M},w}$ conjuncts which

are the result of replacing A_l by B_l and n by $n+1$ in (2) and (3). Similar to our notation for the A-counter, we denote by $(B = k)$ the concept that expresses that the B-counter has the value k. We now use

$$\Box\big(\top \sqsubseteq \exists r.(B = 0)\big) \wedge \Box\big(\exists r.\bigcirc(B = 0) \sqsubseteq \bot\big)$$

to ensure that

- at each time point, every element has an r-successor encoding the "start" of the B-counter, and
- this r-connection "starts" to exist exactly when the B-counter starts (and exists for at least $2^n + 1$ and at most 2^{n+1} time points afterwards).

This way, each domain element can distinguish between 2^{n+1} different r-successors using their B-counter value, which is later also used for the matcher objects. We use the formula

$$\Box\big((B = 2^n) \sqsubseteq N_{\mathsf{Bit}} \leftrightarrow C_{\mathsf{Bit}}\big) \wedge \Box\big(N_{\mathsf{Bit}} \sqcap \bigcirc(B = 0) \sqsubseteq \bot\big) \wedge \Box\big(N_{\mathsf{Bit}} \sqcap \bigcirc(B = 2^n + 1) \sqsubseteq \bot\big)$$

to bring to the current time point the corresponding bit value in the next tape address. The following formulae increment our double exponential counter:

$$\Box\Big(\bigcirc(C_{\mathsf{Bit}}\,\mathcal{U}\mathsf{Tape}) \rightarrow \forall r.\big((B = 0) \rightarrow (C_{\mathsf{Bit}} \leftrightarrow \neg N_{\mathsf{Bit}})\big)\Big)(a),$$
$$\Box\Big(\neg\bigcirc(C_{\mathsf{Bit}}\,\mathcal{U}\mathsf{Tape}) \rightarrow \forall r.\big((B = 0) \rightarrow (C_{\mathsf{Bit}} \leftrightarrow N_{\mathsf{Bit}})\big)\Big)(a).$$

Intuitively, these formulae express that the current bit is flipped iff all following bits are true (until the next time point where Tape holds).

We now implement the matcher objects that are used to transfer information between successive configurations. Each matcher object has an associated address, the bit values of which are represented using the concept C'_{Bit}, which is made rigid using the following formula:

$$\Box(C'_{\mathsf{Bit}} \equiv \bigcirc C'_{\mathsf{Bit}}).$$

The different bits of the address are stored using 2^n r-successors of the matcher object, which are differentiated using their B-counter values. We make sure that the value of C'_{Bit} is uniquely determined by the B-counter value of an r-successor:

$$\Box\Big(\exists r.\big(C'_{\mathsf{Bit}} \sqcap (B = 0)\big) \sqsubseteq \forall r.\big((B = 0) \rightarrow C'_{\mathsf{Bit}}\big)\Big).$$

The following formula now ensures that we have a matcher object for each tape address, and that this domain element is marked with the concept Match if it corresponds to the current tape address:

$$\Box\neg(\mathsf{Match} \sqsubseteq \bot) \wedge \Box\big(\mathsf{Match} \equiv \bigcirc(\mathsf{Match} \sqcup \mathsf{Tape}) \sqcap \exists r.((B = 0) \sqcap (C_{\mathsf{Bit}} \leftrightarrow C'_{\mathsf{Bit}}))\big).$$

Since r is only $2^n + 1$-rigid, we have to make sure that the address remains associated to the matcher object during successive configurations, which is done by the following formulae:

$$\Box\big(\exists r.((B = 2^n) \sqcap C'_{\mathsf{Bit}}) \sqsubseteq \exists r.((B = 0) \sqcap C'_{\mathsf{Bit}})\big),$$
$$\Box\big(\exists r.((B = 2^n) \sqcap \neg C'_{\mathsf{Bit}}) \sqsubseteq \exists r.((B = 0) \sqcap \neg C'_{\mathsf{Bit}})\big).$$

After we have implemented the required tools, we can now describe the actual mechanism of the Turing machine. We encode that all elements of the domain share the information about the tape content and state in the time points marked with the concept name Tape:

$$\Box \bigvee_{\sigma \in \Gamma} \Big(\mathsf{Tape} \sqsubseteq C_\sigma \sqcap \prod_{\tau \in \Gamma \setminus \{\sigma\}} \neg C_\tau\Big),$$

$$\Box \bigvee_{q \in Q'} \Big(\mathsf{Tape} \sqsubseteq C_q \sqcap \prod_{p \in Q' \setminus \{q\}} \neg C_p \Big).$$

We use auxiliary concept names N_p, N_σ to bring to the current tape cell the state and tape content of the next tape cell. In addition, we check, using the concept name Init, whether we have reached the end of the tape, i.e., if the next tape cell is the beginning of the next configuration:

$$\Box\Big((\mathsf{Tape} \sqcap N_\sigma) \leftrightarrow \bigcirc\big(\neg\mathsf{Tape}\, \mathcal{U}(\mathsf{Tape} \sqcap C_\sigma \sqcap \neg\mathsf{Init})\big)\Big)(a),$$

$$\Box\Big((\mathsf{Tape} \sqcap N_q) \leftrightarrow \bigcirc\big(\neg\mathsf{Tape}\, \mathcal{U}(\mathsf{Tape} \sqcap C_q \sqcap \neg\mathsf{Init})\big)\Big)(a).$$

We also have concept names C'_p, C'_σ and N'_p, N'_σ, which are used by the matcher objects to carry information to the next configuration. Whenever there is a match, we synchronise the state and tape content of the corresponding matcher object with the information of the current tape cell. For this, we use the axioms

$$\Box(\mathsf{Tape} \sqcap \mathsf{Match} \sqsubseteq Y \leftrightarrow Y'), \tag{4}$$

where Y ranges over $\mathcal{S} := \{N_\sigma, C_\sigma \mid \sigma \in \Gamma\} \cup \{N_q, C_q \mid q \in Q'\}$. We made sure that these concept names are always satisfied by all or none of the domain elements in the current time point, so that the matcher object satisfying Match has access to this information.

This concludes the main technical setup of our reduction. We now encode the transitions in Θ as follows.

For $\Theta(q,\sigma) = (p,\tau,l)$:	$\Box\big(\mathsf{Tape} \sqcap \mathsf{Match} \sqcap N'_q \sqcap N'_\sigma \sqsubseteq \bigcirc(C'_p \sqcap N'_\tau)\big)$.
For $\Theta(q,\sigma) = (p,\tau,r)$:	$\Box\big(\mathsf{Tape} \sqcap \mathsf{Match} \sqcap C'_q \sqcap C'_\sigma \sqsubseteq \bigcirc(N'_p \sqcap C'_\tau)\big)$.
For $\overline{q}$ and $\sigma \in \Gamma$:	$\Box\big(\mathsf{Tape} \sqcap \mathsf{Match} \sqcap C'_{\overline{q}} \sqcap C'_\sigma \sqsubseteq \bigcirc C'_\sigma\big)$.

The matcher object for the current tape cell thus contains the necessary information for the next configuration at the next time point. To actually propagate this information to the next configuration, we use the axioms

$$\Box\big(Y' \sqsubseteq \bigcirc Y' \sqcup (\mathsf{Tape} \sqcap \mathsf{Match})\big),$$

where Y ranges over $\mathcal{S}$. As soon as the matcher object satisfies Match again, this information is synchronised to the current tape cell using Axiom (4).

It remains to ensure that the non-head worlds are labelled with $\overline{q}$. For this, we mark the beginning of a configuration with the concept name Init and the position of the head with the concept name Head:

$$\Box\Big(\mathsf{Init} \leftrightarrow \big(\mathsf{Tape} \sqcap \neg C_{\mathsf{Bit}} \sqcap \bigcirc(\neg C_{\mathsf{Bit}}\, \mathcal{U}\mathsf{Tape})\big)\Big)(a),$$

$$\Box\Big(\mathsf{Init} \rightarrow \big(\mathsf{Head} \sqcup \bigcirc(\neg\mathsf{Init}\, \mathcal{U}\mathsf{Head})\big)\Big)(a),$$

$$\Box\Big(\mathsf{Head} \leftrightarrow (\neg C_{\overline{q}} \sqcap \mathsf{Tape})\Big)(a),$$

$$\Box\Big(\mathsf{Head} \rightarrow \bigcirc(\neg\mathsf{Head}\, \mathcal{U}\mathsf{Init})\Big)(a).$$

We now encode the initial configuration. Let $w = \sigma_1 \cdots \sigma_m$ be the input word for $\mathcal{M}$. We define the shorthand $(\varphi\, \mathcal{U}^i \gamma, \chi)$ inductively by setting $(\varphi\, \mathcal{U}^1 \gamma, \chi) := \varphi\, \mathcal{U}(\gamma \sqcap \chi)$ and $\varphi\, \mathcal{U}(\gamma \sqcap \bigcirc(\varphi\, \mathcal{U}^{i-1}\gamma, \chi))$ for $i > 1$. We use the axioms

$$\Big(C_{q_0} \sqcap C_{\sigma_1} \sqcap \prod_{i=1}^{m-1} \bigcirc\big(\neg\mathsf{Tape}\, \mathcal{U}^i(\mathsf{Tape}, C_{\sigma_{i+1}})\big)\Big)(a),$$

$$\bigcirc\big(\neg\mathsf{Tape}\,\mathcal{U}^m(\mathsf{Tape}, C_b\,\mathcal{U}\mathsf{Init})\big)(a),$$
$$\Diamond \bigvee_{q\in F} (\mathsf{Tape}\sqcap C_q)(a).$$

Intuitively, C_{σ_1} and the big conjunction enforce that the input word is written on the tape, and $C_b\,\mathcal{U}\mathsf{Init}$ ensures that the remaining cells are labelled with the blank symbol. Finally, the last axiom expresses that a final state is reachable.

This finishes the description of $\varphi_{\mathcal{M},w}$. Given this construction, one can show that $\mathcal{M}$ accepts w iff $\varphi_{\mathcal{M},w}$ is satisfiable. □

REFERENCES

[1] Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. 1996. The Benefits of Relaxing Punctuality. *J. ACM* 43, 1 (1996), 116–146. https://doi.org/10.1145/227595.227602

[2] Rajeev Alur and Thomas A. Henzinger. 1993. Real-Time Logics: Complexity and Expressiveness. *Inf. Comput.* 104, 1 (1993), 35–77. https://doi.org/10.1006/inco.1993.1025

[3] Rajeev Alur and Thomas A. Henzinger. 1994. A Really Temporal Logic. *J. ACM* 41, 1 (1994), 181–204. https://doi.org/10.1145/174644.174651

[4] Alessandro Artale, Davide Bresolin, Angelo Montanari, Guido Sciavicco, and Vladislav Ryzhikov. 2014. DL-Lite and Interval Temporal Logics: A Marriage Proposal. In *Proc. of the 21st Eur. Conf. on Artificial Intelligence (ECAI'14) (Frontiers in Artificial Intelligence and Applications, Vol. 263)*, Torsten Schaub (Ed.). IOS Press, 957–958. https://doi.org/10.3233/978-1-61499-419-0-957

[5] Alessandro Artale, Roman Kontchakov, Alisa Kovtunova, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. 2015. First-Order Rewritability of Ontology-Mediated Temporal Queries. In *Proc. of the 24th Int. Joint Conf. on Artificial Intelligence (IJCAI'15)*, Qiang Yang and Michael Wooldridge (Eds.). AAAI Press, 2706–2712. http://ijcai.org/Abstract/15/383

[6] Alessandro Artale, Roman Kontchakov, Alisa Kovtunova, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. 2017. Ontology-Mediated Query Answering over Temporal Data: A Survey. In *Proc. of the 24th Int. Symp. on Temporal Representation and Reasoning (TIME'17) (Leibniz International Proceedings in Informatics, Vol. 90)*, Sven Schewe, Thomas Schneider, and Jef Wijsen (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 1:1–1:37. https://doi.org/10.4230/LIPIcs.TIME.2017.1

[7] Alessandro Artale, Roman Kontchakov, Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. 2007. Temporalising Tractable Description Logics. In *Proc. of the 14th Int. Symp. on Temporal Representation and Reasoning (TIME'07)*, Valentin Goranko and X. Sean Wang (Eds.). IEEE Press, 11–22. https://doi.org/10.1109/TIME.2007.62

[8] Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. 2014. A Cookbook for Temporal Conceptual Data Modelling with Description Logics. *ACM Trans. Comput. Log.* 15, 3 (2014), 25. https://doi.org/10.1145/2629565

[9] Alessando Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. 2015. Tractable Interval Temporal Propositional and Description Logics. In *Proc. of the 29th AAAI Conf. on Artificial Intelligence (AAAI'15)*, Blai Bonet and Sven Koenig (Eds.). AAAI Press, 1417–1423. https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9638

[10] Alessandro Artale, Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. 2013. Temporal Description Logic for Ontology-Based Data Access. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence (IJCAI'13)*, Francesca Rossi (Ed.). AAAI Press, 711–717. http://ijcai.org/Abstract/13/112

[11] Alessandro Artale, Carsten Lutz, and David Toman. 2007. A Description Logic of Change. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI'07)*, Manuela M. Veloso (Ed.). IJCAI, 218–223. http://www.ijcai.org/Proceedings/07/Papers/033.pdf

[12] Alessandro Artale, Andrea Mazzullo, and Ana Ozaki. 2018. Temporal Description Logics over Finite Traces. In *Proc. of the 31st Int. Workshop on Description Logics (DL'18) (CEUR Workshop Proceedings, Vol. 2211)*, Magdalena Ortiz and Thomas Schneider (Eds.). http://ceur-ws.org/Vol-2211/paper-06.pdf

[13] Alessandro Artale, Andrea Mazzullo, and Ana Ozaki. 2019. Do You Need Infinite Time?. In *Proc. of the 28th Int. Joint Conf. on Artificial Intelligence (IJCAI'19)*, Sarit Kraus (Ed.). ijcai.org, 1516–1522. https://doi.org/10.24963/ijcai.2019/210

[14] Alessandro Artale, Andrea Mazzullo, and Ana Ozaki. 2019. Temporal DL-Lite over Finite Traces (Preliminary Results). In *Proc. of the 32nd Int. Workshop on Description Logics (DL'19) (CEUR Workshop Proceedings, Vol. 2373)*, Mantas Simkus and Grant E. Weddell (Eds.). http://ceur-ws.org/Vol-2373/paper-2.pdf

[15] Franz Baader, Stefan Borgwardt, and Walter Forkel. 2018. Patient Selection for Clinical Trials Using Temporalized Ontology-Mediated Query Answering. In *Proc. of the 1st Int. Workshop on Hybrid Question Answering with Structured and Unstructured Knowledge (HQA'18), Companion Volume of WWW'18*, Franz Baader, Brigitte Grau, and Yue Ma (Eds.). ACM, 1069–1074. https://doi.org/10.1145/3184558.3191538

[16] Franz Baader, Stefan Borgwardt, Patrick Koopmann, Ana Ozaki, and Veronika Thost. 2017. Metric Temporal Description Logics with Interval-Rigid Names. In *Proc. of the 11th Int. Symp. on Frontiers of Combining Systems (FroCoS'17) (Lecture Notes in Computer Science, Vol. 10483)*, Clare Dixon and Marcelo Finger (Eds.). Springer-Verlag, 60–76. https://doi.org/10.1007/978-3-319-66167-4_4

[17] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. 2015. Temporal Conjunctive Queries in Expressive Description Logics with Transitive Roles. In *Proc. of the 28th Australasian Joint Conf. on Artificial Intelligence (AI'15) (Lecture Notes in Artificial Intelligence, Vol. 9457)*, Bernhard Pfahringer and Jochen Renz (Eds.). Springer-Verlag, 21–33. https://doi.org/10.1007/978-3-319-26350-2_3

[18] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. 2015. Temporal Query Entailment in the Description Logic $\mathcal{SHQ}$. *J. Web Semant.* 33 (2015), 71–93. https://doi.org/10.1016/j.websem.2014.11.008

[19] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider (Eds.). 2007. *The Description Logic Handbook: Theory, Implementation, and Applications* (2nd ed.). Cambridge University Press.

[20] Franz Baader, Silvio Ghilardi, and Carsten Lutz. 2012. LTL over Description Logic Axioms. *ACM Trans. Comput. Log.* 13, 3 (2012), 21:1–21:32. https://doi.org/10.1145/2287718.2287721

[21] Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. 2015. Temporalizing Rewritable Query Languages over Knowledge Bases. *J. Web Semant.* 33 (2015), 50–70. https://doi.org/10.1016/j.websem.2014.11.007

[22] Stefan Borgwardt and Veronika Thost. 2015. Temporal Query Answering in *DL-Lite* with Negation. In *Proc. of the 1st Global Conf. on Artificial Intelligence (GCAI'15) (EasyChair Proceedings in Computing, Vol. 36)*, Georg Gottlob, Geoff Sutcliffe, and Andrei Voronkov (Eds.). EasyChair, 51–65. https://easychair.org/publications/paper/T8jq

[23] Stefan Borgwardt and Veronika Thost. 2015. Temporal Query Answering in the Description Logic $\mathcal{EL}$. In *Proc. of the 24th Int. Joint Conf. on Artificial Intelligence (IJCAI'15)*, Qiang Yang and Michael Wooldridge (Eds.). AAAI Press, 2819–2825. http://ijcai.org/Abstract/15/399

[24] Patricia Bouyer, Nicolas Markey, Joël Ouaknine, and James Worrell. 2007. The Cost of Punctuality. In *Proc. of the 22nd IEEE Symp. on Logic in Computer Science (LICS'07)*. IEEE Press, 109–120. https://doi.org/10.1109/LICS.2007.49

[25] Sebastian Brandt, Elem Güzel Kalaycı, Roman Kontchakov, Vladislav Ryzhikov, Guohui Xiao, and Michael Zakharyaschev. 2017. Ontology-Based Data Access with a Horn Fragment of Metric Temporal Logic. In *Proc. of the 31st AAAI Conf. on Artificial Intelligence (AAAI'17)*, Satinder Singh and Shaul Markovitch (Eds.). AAAI Press, 1070–1076. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14881

[26] Sebastian Brandt, Elem Güzel Kalaycı, Vladislav Ryzhikov, Guohui Xiao, and Michael Zakharyaschev. 2018. Querying Log Data with Metric Temporal Logic. *J. Artif. Intell. Res.* 62 (2018), 829–877. https://doi.org/10.1613/jair.1.11229

[27] Christopher L. Crowe and Cui Tao. 2015. Designing Ontology-based Patterns for the Representation of the Time-Relevant Eligibility Criteria of Clinical Protocols. *AMIA Summits on Translational Science Proceedings* 2015 (2015), 173–177. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4525239/

[28] E. Allen Emerson, Aloysius K. Mok, A. Prasad Sistla, and Jai Srinivasan. 1992. Quantitative Temporal Reasoning. *Real-Time Syst.* 4 (1992), 331–352. https://doi.org/10.1007/BF00355298

[29] Carlo A. Furia and Paola Spoletini. 2008. Tomorrow and All our Yesterdays: MTL Satisfiability over the Integers. In *Proc. of the 5th Int. Coll. on Theoretical Aspects of Computing (ICTAC'08) (Lecture Notes in Computer Science, Vol. 5160)*. Springer-Verlag, 126–140. https://doi.org/10.1007/978-3-540-85762-4_9

[30] Dov M. Gabbay, Agi Kurucz, Frank Wolter, and Michael Zakharyaschev. 2003. *Many-dimensional modal logics: Theory and applications*. Gulf Professional Publishing.

[31] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Roman Kontchakov. 2016. Temporalized $\mathcal{EL}$ Ontologies for Accessing Temporal Data: Complexity of Atomic Queries. In *Proc. of the 25th Int. Joint Conf. on Artificial Intelligence (IJCAI'16)*, Subbarao Kambhampati (Ed.). AAAI Press, 1102–1108. https://www.ijcai.org/Abstract/16/160

[32] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Ana Ozaki. 2016. On Metric Temporal Description Logics. In *Proc. of the 22nd Eur. Conf. on Artificial Intelligence (ECAI'16) (Frontiers in Artificial Intelligence and Applications, Vol. 285)*, Gal A. Kaminka and Maria Fox (Eds.). IOS Press, 837–845. https://doi.org/10.3233/978-1-61499-672-9-837

[33] Joseph Y. Halpern and Moshe Y. Vardi. 1989. The Complexity of Reasoning about Knowledge and Time. I. Lower Bounds. *J. Comput. Syst. Sci.* 38, 1 (1989), 195–237. https://doi.org/10.1016/0022-0000(89)90039-1

[34] David Harel. 1986. Effective transformations on infinite trees, with applications to high undecidability, dominoes, and fairness. *J. ACM* 33, 1 (1986), 224–248. https://doi.org/10.1145/4904.4993

[35] Ullrich Hustadt, Ana Ozaki, and Clare Dixon. 2017. Theorem Proving for Metric Temporal Logic over the Naturals. In *Proc. of the 26th Int. Conf. on Automated Deduction (CADE'17) (Lecture Notes in Computer Science, Vol. 10395)*, Leonardo de Moura (Ed.). Springer-Verlag, 326–343. https://doi.org/10.1007/978-3-319-63046-5_20

[36] Patrick Koopmann. 2019. Ontology-Based Query Answering for Probabilistic Temporal Data. In *Proc. of the 33rd AAAI Conf. on Artificial Intelligence (AAAI'19)*, Pascal Van Hentenryck and Zhi-Hua Zhou (Eds.). AAAI Press, 2903–2910.

[37] François Laroussinie, Nicolas Markey, and Philippe Schnoebelen. 2002. Temporal Logic with Forgettable Past. In *Proc. of the 17th Annual IEEE Symp. on Logic in Computer Science (LICS'02)*. IEEE Press, 383–392. https://doi.org/10.1109/LICS.2002.1029846

[38] Carsten Lutz, Dirk Walther, and Frank Wolter. 2007. Quantitative Temporal Logics over the Reals: PSPACE and Below. *Inf. Comput.* 205, 1 (2007), 99–123. https://doi.org/10.1016/j.ic.2006.08.006

[39] Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. 2008. Temporal Description Logics: A Survey. In *Proc. of the 15th Int. Symp. on Temporal Representation and Reasoning (TIME'08)*, Stéphane Demri and Christian S. Jensen (Eds.). IEEE Press, 3–14. https://doi.org/10.1109/TIME.2008.14

[40] Joël Ouaknine and James Worrell. 2008. Some Recent Results in Metric Temporal Logic. In *Proc. of the 6th Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS'08) (Lecture Notes in Computer Science, Vol. 5215)*. Springer-Verlag, 1–13. https://doi.org/10.1007/978-3-540-85778-5_1

[41] Amir Pnueli. 1977. The Temporal Logic of Programs. In *Proc. of the 18th Annual Symp. on Foundations of Computer Science (SFCS'77)*. IEEE Computer Society, 46–57. https://doi.org/10.1109/SFCS.1977.32

[42] Morteza Yousef Sanati. 2015. *A Metric Interval-Based Temporal Description Logic.* PhD Thesis. McMaster University, Hamilton, Canada. http://hdl.handle.net/11375/16783

[43] Klaus Schild. 1991. A Correspondence Theory for Terminological Logics: Preliminary Report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, John Mylopoulos and Raymond Reiter (Eds.). Morgan Kaufmann, 466–471. http://ijcai.org/Proceedings/91-1/Papers/072.pdf

[44] Stefan Schulz, Kornél Markó, and Bontawee Suntisrivaraporn. 2008. Formal Representation of Complex SNOMED CT Expressions. *BMC Med. Inform. Decis.* 8, Suppl 1 (2008), S9. https://doi.org/10.1186/1472-6947-8-S1-S9

[45] A. Prasad Sistla and Edmund M. Clarke. 1985. The Complexity of Propositional Linear Temporal Logics. *J. ACM* 32, 3 (1985), 733–749. https://doi.org/10.1145/3828.3837

[46] Veronika Thost. 2018. Metric Temporal Extensions of *DL-Lite* and Interval-Rigid Names. In *Proc. of the 16th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'18)*, Frank Wolter, Michael Thielscher, and Francesca Toni (Eds.). AAAI Press, 665–666. https://aaai.org/ocs/index.php/KR/KR18/paper/view/18035 Extended abstract.

[47] Przemysław Andrzej Wałęga, Mark Kaminski, and Bernardo Cuenca Grau. 2019. Reasoning over Streaming Data in Metric Temporal Datalog. In *Proc. of the 33rd AAAI Conference on Artificial Intelligence (AAAI'19)*, Pascal Van Hentenryck and Zhi-Hua Zhou (Eds.). AAAI Press, 3092–3099. https://doi.org/10.1609/aaai.v33i01.33013092

[48] Frank Wolter and Michael Zakharyaschev. 2000. Temporalizing Description Logics. In *Frontiers of Combining Systems 2 (Studies in Logic and Computation, Vol. 7)*, Dov Gabbay and Maarten de Rijke (Eds.). Research Studies Press/Wiley, 379–402.