ECAI 2016
G.A. Kaminka et al. (Eds.)

doi:10.3233/978-1-61499-672-9-837

On Metric Temporal Description Logics

Víctor Gutiérrez-Basulto[1] and **Jean Christoph Jung**[1] and **Ana Ozaki**[2]

Abstract. We introduce *metric* temporal description logics (*m*TDLs) as combinations of the classical description logic $\mathcal{ALC}$ with (*a*) LTL$^{\mathsf{bin}}$, an extension of the temporal logic LTL with succinctly represented intervals, and (*b*) metric temporal logic MTL, extending LTL$^{\mathsf{bin}}$ with capabilities to quantitatively reason about time delays. Our main contributions are algorithms and tight complexity bounds for the satisfiability problem in these *m*TDLs: For *m*TDLs based on (fragments of) LTL$^{\mathsf{bin}}$, we establish complexity bounds ranging from ExpTime to 2ExpSpace. For *m*TDLs based on (fragments of) MTL interpreted over the naturals, we establish complexity bounds ranging from ExpSpace to 2ExpSpace.

1 Introduction

Classical Description logics (DLs) are fragments of first-order logic aiming at the representation of and reasoning about knowledge. The importance of DLs lies in the fact that they are, arguably, the prime formalism to encode ontologies, e.g., they underpin the web ontology language OWL 2, the medical ontology SNOMED CT and the thesaurus of the US national cancer institute. It has been observed that in many domains where ontologies are used an implicit or explicit notion of *time* plays a central role [25]. As an instance, many terms in the medical domain are described making reference to temporal patterns; for example, the description of the autoimmune disease diabetes must specify that it might lead to glaucoma in the future. On the other hand, DLs were initially developed with the aim of capturing *static* knowledge. To address this shortcoming, over the last 20 years a plethora of temporal DLs (TDLs), extensions of DLs with an explicit temporal component, have been proposed [5, 25].

The most popular approach to constructing TDLs is to combine classical DLs with temporal logics such as LTL and CTL, and to provide a two-dimensional product-like semantics [29, 15, 25]–one dimension for time and the other for DL quantification. Temporal DLs of this kind support the definition of terms using, e.g., the temporal operators 'at the next/previous point' or 'somewhere in the future/past'. As an example, in TDLs based on CTL we can use $\exists \mathsf{has.Diabetes} \sqsubseteq \mathbf{E}\Diamond\exists \mathsf{develops.Glaucoma}$ to say that 'a patient with diabetes may develop glaucoma in the future'. The importance of TDLs based on LTL and CTL is witnessed by the vast amount of research conducted on the topic in the last decade; in particular, TDLs using expressive as well as lightweight DLs, with different levels of interaction between the components, have been investigated [31, 7, 25, 13, 17, 8, 18, 19]. Moreover, this sort of TDLs have been already successfully used in applications, e.g., to describe conceptual models capturing the evolution of databases over time [8]. However, their temporal constructs does not seem to always adequately carter for the needs of ontology designers and users. Indeed, temporal primitives such as 'eventually in the future' might not be sufficiently precise for temporal conceptual modeling in an ontology. As an instance, in the medical domain ontological modelling often requires reference to concrete durations. Consider, for example, lyme disease: Affected patients develop a rash within 3-32 days after the infection. Since the infection can only occur after exposure to ticks, the concrete temporal interval of 3-32 days can be used to rule out lyme disease as a cause of rash.

Observe that, although it is possible to express eventuality within an interval by 'unfolding' all the timepoints represented in an interval, allowing intervals in the language with end-points in *binary* would result in an exponentially more succinct statement. For instance, for expressing that a patient "develops a rash eventually within 3-32 days", in TDLs based on plain LTL, the ontology designer has to write $\bigcirc^3(\exists \mathsf{has.Rash}) \sqcup \ldots \sqcup \bigcirc^{32}(\exists \mathsf{has.Rash})$, whereas the same can be expressed elegantly, and more succinctly by $\Diamond_{[3,32]}\exists \mathsf{has.Rash}$ in the logics studied in this paper. Despite the need of this feature, TDLs based on temporal logics succinctly capturing time intervals (to the best of our knowledge) have not yet been considered in the literature. It is important to note that TDLs based on Halpern and Shoham's ($\mathcal{HS}$) interval logic of Allen's relations have been recently investigated [3, 9]. However, these TDLs are orthogonal to the ones investigated here because they are interval-based logics, i.e., intervals (instead of time points) are the basic time units.

The purpose of this paper is to initiate the study of metric TDLs (*m*TDLs) allowing for quantitative temporal reasoning. In particular, we are interested in TDLs merging qualitative temporal assertions together with quantitative constraints so as to get the benefits of the qualitative and quantitative abstraction levels. To this end, we consider TDLs based on (*a*) LTL$^{\mathsf{bin}}$, the extension of LTL with succinctly represented intervals, and (*b*) the real-time metric temporal logic MTL, extending LTL$^{\mathsf{bin}}$ with capabilities to quantitatively reason about time delays. We look at TDLs that might prove useful for applications: we consider the traditional DL $\mathcal{ALC}$, make the (most-general) constant domain assumption, and apply temporal operators to concepts and, in the second part of the paper, to TBox statements. We do not apply temporal operators to roles since this typically leads to undecidability [25]. Our main contributions are algorithms for the satisfiability problem and complexity bounds.

Our study starts with *m*TDLs based on LTL$^{\mathsf{bin}}$ and temporal operators applied only to concepts. For example, in LTL$^{\mathsf{bin}}_{\mathcal{ALC}}$ we can use

$$\exists \mathsf{exposedTo.Tick} \sqsubseteq \Box_{[3,32]}(\exists \mathsf{has.Rash} \rightarrow \Diamond_{[0,3]}\exists \mathsf{gets.LymeDTest})$$

to say that 'persons exposed to ticks whom develop a rash within 3-32 days after that must be tested for Lyme disease within three days.'

We have argued above that LTL$^{\mathsf{bin}}_{\mathcal{ALC}}$ is not more expressive than LTL$_{\mathcal{ALC}}$; however, the (exponential!) translation does not give tight complexity bounds. More specifically, the translation yields a 2Ex-

[1] University of Bremen, Germany; {victor,jeanjung}@tzi.de
[2] University of Liverpool, UK; anaozaki@liverpool.ac.uk

	propositional	combined with $\mathcal{ALC}$ *temporal operators on concepts*	combined with $\mathcal{ALC}$ *temporal operators on concepts and TBoxes*
LTL	PSPACE [30]	EXPTIME [25]	EXPSPACE [15]
LTL^{bin}	EXPSPACE [1]	EXPSPACE [Thm. 1]	2EXPSPACE [Thm. 5]
$\text{LTL}^{0,\infty}$	PSPACE [24]	EXPTIME [Thm. 2]	EXPSPACE [Thm. 7]
MTL	EXPSPACE [1]	2EXPSPACE [Thm. 3]	2EXPSPACE [Thm. 6]
$\text{MTL}^{0,\infty}$	PSPACE [24]	EXPSPACE [Thm. 4]	EXPSPACE [Thm. 8]

Table 1: Overview of previous and new complexity results.

PTIME upper bound because satisfiability in $\text{LTL}_{\mathcal{ALC}}$ can be done in EXPTIME [25]. In contrast, in our first main result, we develop an algorithm based on quasimodels in order to obtain EXPSPACE-completeness for satisfiability in $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$. Recall that satisfiability in LTL^{bin} is also EXPSPACE-complete [1], thus, the combination with $\mathcal{ALC}$ is "for free."

As the next step, we consider as temporal component metric temporal logic MTL [23], which extends LTL^{bin} by explicitly associating to each state a timestamp, allowing then for quantitative reasoning of time-delays. MTL has been intensively studied in the last 20 years; in particular, different semantics, syntactic restrictions and underlying time domains have been considered, for an overview see [27]. We will consider here MTL over the naturals with so-called *pointwise semantics*. Under this semantics we can see states as *observations*, say of a real-time system, that have an explicit discrete timestamp and consequently think of the time difference between two consecutive observations as a time-delay. For example, in $\text{MTL}_{\mathcal{ALC}}$ we use

$$\mathsf{PhDStudent} \sqcap \neg\exists\mathsf{pays.Fees} \sqsubseteq \bigcirc_{[1,3]}(\exists\mathsf{gets.Reminder} \sqcap (\Diamond_{[0,7]}\exists\mathsf{pays.Fees} \sqcup (\neg\exists\mathsf{access.Lab}\, \mathcal{U}_{[7,\infty)}\exists\mathsf{pays.Fees}))$$

to express that, if the system observes that a PhD student has not paid the fees, then it should issue a reminder in the next system cycle (that is, observation) which is necessarily occurring within the next three time units; moreover, the student then should pay the fees within seven time units (for example days) or she does not have access to the lab until she pays.

Based on the aforementioned result for $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ and the limited interaction of the dimensions, one could conjecture that the complexity of $\text{MTL}_{\mathcal{ALC}}$ is not higher than that of the components, and therefore EXPSPACE-complete. Surprisingly, we show that this is not the case by establishing a 2EXPSPACE lower bound, which is later shown to be tight.

We then turn our attention to the case where temporal operators are additionally applied to TBoxes. As for the basic case, we start by looking at $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$. Most interestingly, it can be shown that the aforementioned 2EXPSPACE-hardness result can be lifted to $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ temporal TBox satisfiabilty. Matching upper bounds for $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ and $\text{MTL}_{\mathcal{ALC}}$ follow from the translation to the qualitative case $\text{LTL}_{\mathcal{ALC}}$, where TBox satisifiability is known to be EXPSPACE-complete [25].

Based on similar observations for the propositional case [24], we finally looked at the restrictions $\text{LTL}^{0,\infty}_{\mathcal{ALC}}$ and $\text{MTL}^{0,\infty}_{\mathcal{ALC}}$ where intervals are only of the form $[0, c]$ or $[c, \infty]$. This is still expressive enough to succinctly model, for instance, time limits. We show that, indeed, the quasimodel technique can be leveraged to show that this leads to better complexity in many cases.

An overview of existing and new results is given in Table 1.

Missing proofs are provided in an extended version, available at www.informatik.uni-bremen.de/tdki/research/papers/GJO16.pdf.

2 Preliminaries

Intervals. We use standard notation for (open and closed) intervals, e.g., $[c_1, c_2)$ is the set of all $n \in \mathbb{N}$ with $c_1 \leq n < c_2$. It is thus clear what is meant with $n \in I$ and $I \subseteq I'$ for intervals I, I'.

$\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ **syntax.** $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ is a TDL based on LTL and the classical DL $\mathcal{ALC}$. Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be countably infinite sets of *concept* and *role names*, respectively. $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$*-concepts* C, D are formed according to the rule:

$$C, D ::= A \mid \neg C \mid C \sqcap D \mid \exists r.C \mid \bigcirc C \mid C\mathcal{U}_I D$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and I is an *interval* of the form $[c_1, c_2]$ or $[c_1, \infty)$ with $c_1, c_2 \in \mathbb{N}$ given in *binary*. We use standard Boolean and temporal abbreviations: $C \sqcup D$, $\forall r.C$, $\top$, $\Diamond_I C$, and $\Box_I C$ for $\neg(\neg C \sqcap \neg D)$, $\neg\exists r.\neg C$, $A \sqcup \neg A$, $\top\mathcal{U}_I C$, and $\neg\Diamond_I \neg C$, respectively. We omit intervals of the form $[0, \infty)$ and write $C\mathcal{U}D$ instead of $C\mathcal{U}_{[0,\infty)}D$, and use the subscript $\cdot_c$ to refer to intervals of the form $[c, c]$.

An $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ *TBox* is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$ with C, D $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-concepts. We use $C \equiv D$ to refer to the two concept inclusions $C \sqsubseteq D$ and $D \sqsubseteq C$. The *size* of a TBox $\mathcal{T}$ (a concept C) is the number of symbols required to write $\mathcal{T}$ (C).

$\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ **semantics**. The semantics of $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ is given in terms of *interpretations*, that is, structures $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_n)_{n\in\mathbb{N}})$, where each $\mathcal{I}_n$ is a classical DL interpretation with domain $\Delta^{\mathfrak{I}}$: we have $A^{\mathcal{I}_n} \subseteq \Delta^{\mathfrak{I}}$ and $r^{\mathcal{I}_n} \subseteq \Delta^{\mathfrak{I}} \times \Delta^{\mathfrak{I}}$. We usually write $A^{\mathfrak{I},n}$ and $r^{\mathfrak{I},n}$ instead of $A^{\mathcal{I}_n}$ and $r^{\mathcal{I}_n}$, respectively. For instance, $d \in A^{\mathfrak{I},n}$ means that in the interpretation $\mathfrak{I}$, the object d is an instance of the concept name A at *time point* n. The stipulation that all time points share the same domain $\Delta^{\mathfrak{I}}$ is called the *constant domain assumption* (meaning that objects are not created or destroyed over time), and it is the most general choice in the sense that increasing, decreasing, and varying domains can all be reduced to it [15].

We now define the semantics of $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-concepts. To this end, we extend the mapping $\cdot^{\mathfrak{I},n}$ from concept names to complex $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-concepts as follows:

$$\begin{aligned}
(\neg C)^{\mathfrak{I},n} &= \Delta^{\mathfrak{I}} \setminus C^{\mathfrak{I},n},\\
(C \sqcap D)^{\mathfrak{I},n} &= C^{\mathfrak{I},n} \cap D^{\mathfrak{I},n},\\
(\exists r.C)^{\mathfrak{I},n} &= \{d \in \Delta^{\mathfrak{I}} \mid \exists e \in C^{\mathfrak{I},n} \text{ with } (d, e) \in r^{\mathfrak{I},n}\},\\
(\bigcirc C)^{\mathfrak{I},n} &= \{d \in \Delta^{\mathfrak{I}} \mid d \in C^{\mathfrak{I},n+1}\},\\
(C\mathcal{U}_I D)^{\mathfrak{I},n} &= \{d \in \Delta^{\mathfrak{I}} \mid \exists k > n :\ d \in D^{\mathfrak{I},k} \wedge k - n \in I\\
&\qquad \wedge \forall m \in (n, k) :\ d \in C^{\mathfrak{I},m}\}.
\end{aligned}$$

An interpretation $\mathfrak{I}$ is a *model* of a concept C if $C^{\mathfrak{I},0} \neq \emptyset$; it is a model of a CI $C \sqsubseteq D$, written $\mathfrak{I} \models C \sqsubseteq D$, if $C^{\mathfrak{I},n} \subseteq D^{\mathfrak{I},n}$, for all $n \in \mathbb{N}$. We call $\mathfrak{I}$ a *model of a TBox* $\mathcal{T}$, written $\mathfrak{I} \models \mathcal{T}$, if $\mathfrak{I} \models \alpha$ for all $\alpha \in \mathcal{T}$. Note that TBoxes are interpreted *globally* in the sense that all CIs must be satisfied at every time point.

Reasoning problem. We are interested in the reasoning problem of *satisfiability relative to global TBoxes* (throughout the paper only called satisfiability), that is, given an $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-concept C and TBox $\mathcal{T}$, decide whether C and $\mathcal{T}$ have a common model.

Sequences. Throughout the paper we use sequences with the following notation. For a (possibly infinite) sequence $\sigma = \sigma(0)\sigma(1)\ldots$, we write $\sigma^{\leq n}$ and $\sigma^{>n}$ for the *head* $\sigma(0)\sigma(1)\ldots\sigma(n)$ and *tail* $\sigma(n+1)\sigma(n+2)\ldots$ of σ, respectively. We also write $\sigma^{>i,\leq j}$ for the subsequence $\sigma(i+1)\ldots\sigma(j)$ of σ. For a finite sequence σ_1 and a sequence σ_2, we denote with $\sigma_1 * \sigma_2$, or just $\sigma_1\sigma_2$ if no confusion is possible, the *concatenation* of σ_1 and σ_2. As usual, we define $\sigma^1 = \sigma$, $\sigma^{n+1} = \sigma\sigma^n$ and $\sigma^\omega = \sigma\sigma\sigma\ldots$.

3 $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$

We aim at devising algorithms and establishing tight complexity bounds for the satisifiability problem. We first concentrate on developing an algorithm for satisfiability in $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$, yielding a tight ExpSpace upper bound. The lower bound is a consequence of the following: (i) allowing the abbreviation $\bigcirc^n$ (meaning n consecutive 'next' operators) with n encoded in binary in LTL makes satisfiability checking ExpSpace-hard [1, 2] and (ii) $\bigcirc^n$ can be expressed in $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ with $\Diamond_n$.

In the second part of this section, we show that satisfiability in the restriction $\text{LTL}^{0,\infty}_{\mathcal{ALC}}$ of $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ to intervals of the form $[0, c]$ or $[c, \infty)$ is complexity-wise better-behaved. In particular, it is ExpTime-complete and thus not harder than in $\mathcal{ALC}$.

The main structure underlying our decision procedure are so-called *quasimodels*, which have been used for studying the satisfiability in various TDLs [31, 15, 7, 17]. In a nutshell, a quasimodel is an abstraction of an interpretation $\mathfrak{I} = (\Delta^{\mathfrak{I}}, (\mathcal{I}_n)_{n\in\mathbb{N}})$ in which each (possibly infinite) $\mathcal{I}_n$ is replaced by a *quasistate*, that is, a *finite* set of *types*.

We show that quasimodels exhibit a monotonic behavior and apply regularity arguments to show membership in ExpSpace and ExpTime, respectively. We show that to check satisfiability it suffices to consider quasimodels of the form:

$$S(0)S(1)\ldots S(n)^\omega, \tag{1}$$

with $S(i) \supseteq S(i+1)$, for all $0 \leq i < n$, and n *double-* and *single-exponentially* bounded in the size of C and $\mathcal{T}$ for $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ and $\text{LTL}^{0,\infty}_{\mathcal{ALC}}$, respectively.

Note that a similar regularity condition holds for LTL in the sense that every satisfiable LTL formula has a regular model like (1) (with $S(i)$ propositional valuations) [26]. The main difference is that n is exponentially-bounded (satisfiability is thus PSpace) and that a larger suffix could be the regular part repeating infinitely; in $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$, due to monotonicity, $S(n)$ is the only periodic set.

Throughout the section, we assume without loss of generality that the TBox $\mathcal{T}$ is of the form $\{\top \sqsubseteq C_{\mathcal{T}}\}$ and denote with $\mathsf{sub}(C, \mathcal{T})$ the set of all subconcepts of C and $C_{\mathcal{T}}$.

3.1 Full $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$

We start with introducing some required notation. Denote with $\mathsf{cl}(C, \mathcal{T})$ the closure under single negations of the set:

$$\mathsf{sub}(C, \mathcal{T}) \cup \{D\mathcal{U}E \mid D\mathcal{U}_{[c,\infty)}E \in \mathsf{sub}(C, \mathcal{T})\}. \tag{‡}$$

As usual, a *type for C and $\mathcal{T}$* is a subset $t \subseteq \mathsf{cl}(C, \mathcal{T})$ such that:

- $D \in t$ iff $\neg D \notin t$, for all $\neg D \in \mathsf{cl}(C, \mathcal{T})$;
- $D \sqcap E \in t$ iff $\{D, E\} \subseteq t$, for all $D \sqcap E \in \mathsf{cl}(C, \mathcal{T})$;
- $C_{\mathcal{T}} \in t$.

We will use $\mathsf{tp}(C, \mathcal{T})$ to denote the set of all types for C and $\mathcal{T}$ and $\sharp_{C,\mathcal{T}}$ to denote the number of types, $|\mathsf{tp}(C, \mathcal{T})|$.

We now describe a set of types that appropriately abstracts a classical description logic interpretation $\mathcal{I}_n$. A *quasistate for C and $\mathcal{T}$* is a set $Q \subseteq \mathsf{tp}(C, \mathcal{T})$ of types such that:

- if $t \in Q$ and $\exists r.D \in t$, then there is $t' \in Q$ such that $\{D\} \cup \{\neg E \mid \neg\exists r.E \in t\} \subseteq t'$.

We next show how to temporally relate types in different quasistates; most importantly, regarding how temporal formulas of the form $\bigcirc D$ and $D\mathcal{U}_I E$ are captured. Let $\bar{t} = t(0)t(1)\ldots \in \mathsf{tp}(C, \mathcal{T})$ be a (possibly infinite) sequence of types. We say that $\bar{t}$ *realizes* $D\mathcal{U}_I E$ if there is $m \in I$ such that $E \in t(m)$ and, for all $0 < l < m$, we have $D \in t(l)$. From here on, we use $S = S(0)S(1)\ldots$ to denote an infinite sequence of quasistates for C and $\mathcal{T}$. A *run* $r = r(0)r(1)\ldots$ *through* S is a sequence of types for C and $\mathcal{T}$ such that for all $n \geq 0$:

(R1) $r(n) \in S(n)$;
(R2) $\bigcirc D \in r(n)$ iff $D \in r(n+1)$, for all $\bigcirc D \in \mathsf{cl}(C, \mathcal{T})$;
(R3) $D\mathcal{U}_I E \in r(n)$ iff $r^{\geq n}$ realizes $D\mathcal{U}_I E$, for all $D\mathcal{U}_I E \in \mathsf{cl}(C, \mathcal{T})$.

Intuitively, a run is a sequence of types which characterizes the temporal evolution of a domain element.

We now have the ingredients to formally define a quasimodel. A *quasimodel for C and $\mathcal{T}$* is a pair $(S, \mathfrak{R})$ with $\mathfrak{R}$ a set of runs through S such that:

(Q1) $C \in t$ for some $t \in S(0)$; and
(Q2) for all $t \in S(n)$, $n \geq 0$ there is a run $r \in \mathfrak{R}$ such that $r(n) = t$.

Intuitively, **(Q1)** ensures that C is witnessed at time point 0, and **(Q2)** ensures that each type has an appropriate temporal evolution through the quasimodel. We show in the appendix that concept satisfiability is characterized by the existence of a quasimodel for C and $\mathcal{T}$:

Lemma 1. *There is a model of C and $\mathcal{T}$ iff there is a quasimodel for C and $\mathcal{T}$.*

This characterization, however, does not serve yet as the basis of an algorithm as both S and $\mathfrak{R}$ are infinite. In the next step, we show that quasimodels can be assumed to have a certain regular shape. Henceforth, let K denote the largest constant occurring in C and $\mathcal{T}$ (or 1 if none exist), and let $\ell_1 = (\sharp_{C,\mathcal{T}})^K + K$. We then have the following normal form of quasimodels.

Lemma 2. *There is a quasimodel for C and $\mathcal{T}$ iff there is a quasimodel $(S, \mathfrak{R})$ for C and $\mathcal{T}$ where S is of the form*

$$S = S_0^{n_0} \ldots S_{m-1}^{n_{m-1}} S_m^\omega$$

for quasistates $S_0, \ldots, S_m$ with $S_i \supsetneq S_{i+1}$, $0 \leq i < m$, and numbers $n_0, \ldots, n_{m-1} < \ell_1$.

The proof of Lemma 2 (cf. appendix) proceeds in two steps. (i) We first show that we can extend any quasimodel such that a quasistate at time $i+1$ is contained in the quasistate at time i, that is, $S_{i+1} \subseteq S_i$, $i \geq 0$. (ii) We then show that if ℓ_1 consecutive quasistates coincide, that is, $S(i) = S(i+1) = \ldots = S(i+\ell_1)$ for some $i \geq 0$, then

we can assume that all subsequent quasistates coincide as well, that is, $S(j) = S(i)$ for all $j \geq i$.

Obviously, the (strict!) containment condition on the S_i in Lemma 2 implies that m is at most $\sharp_{C,\mathcal{T}}$ since the $S(i)$ are non-empty sets of types. Moreover, note that, due to ℓ_1, the length of the initial irregular part of S is double-exponentially bounded in the size of C and $\mathcal{T}$. Lemmas 1 and 2 give thus rise to the following non-deterministic procedure for checking concept satisfiability.

1. Non-deterministically choose $m < \sharp_{C,\mathcal{T}}$ non-empty sets of types $S_0 \supsetneq \ldots \supsetneq S_m$ and a sequence $n_0, \ldots, n_{m-1}$ of binary numbers such that $n_i < \ell_1$ for all $0 \leq i < m$.
2. Verify that the sequence S defined as

$$S = S_0^{n_0} \ldots S_{m-1}^{n_{m-1}} S_m^{\omega}$$

can be extended to a quasimodel for C and $\mathcal{T}$, that is, check that:

(a) each S_i, $0 \leq i \leq m$, is a quasistate;

(b) there is a $t \in S_0$ with $C \in t$;

(c) for each $i \geq 0$ and $t \in S(i)$, there is a run r through S such that $r(i) = t$.

The procedure is obviously correct (given Lemmas 1 and 2), but involves a non-effective step: in 2(c), infinitely many tests have to be performed. It thus remains to show how to effectively execute 2(c). To this end, we show, in Lemma 3, that it suffices to check 2(c) for all $i \leq \sharp_{C,\mathcal{T}} \cdot \ell_1$, a double exponential number; then, in Lemma 4, we identify a certain regular form of runs, lending itself to implementation. For both Lemmas, let S be as in Lemma 2.

Lemma 3. *If the condition in 2(c) is satisfied for all $i \leq \sharp_{C,\mathcal{T}} \cdot \ell_1$, then it is satisfied for all $i \geq 0$.*

Proof. Similar to the proof of (ii) in Lemma 2. ❑

Lemma 4. *If there is a r run through S with $r(i) = t$, for some $i \geq 0$, then there is a run r' through S which satisfies $r'(i) = t$ and is of the shape*

$$r' = s(0) \cdots s(k_1) * (s'(0) \cdots s'(k_2))^{\omega}$$

for types $s(0), \ldots, s(k_1), s'(0), \ldots, s'(k_2)$ and $k_1 \leq i + (\sharp_{C,\mathcal{T}})^K$, and $k_2 \leq |\mathsf{cl}(C,\mathcal{T})| \cdot (\sharp_{C,\mathcal{T}})^K$.

Proof. We are going to use the following Claim.

Claim. If r is a run through S and $r^{\geq p-K, \leq p} = r^{\geq q-K, \leq q}$ for some $p < q$, then $r' = r^{\leq p} * r^{>q}$ is a run through S.

Proof of the Claim. We need to show that Conditions **(R1)** to **(R3)** hold for r'. Condition **(R1)** is an immediate consequence of the construction of r' and S. For **(R2)**, we only need to check that for every $\bigcirc D \in \mathsf{cl}(C,\mathcal{T})$, $\bigcirc D \in r'(n)$ iff $D \in r'(n+1)$, for all $n \geq 0$. This follows from the fact that $r(p) = r(q)$ and $r \in \mathfrak{R}$.

For **(R3)**, we check concepts of the form $D\mathcal{U}_I E \in \mathsf{cl}(C,\mathcal{T})$. Note that since $r^{\geq p-K, \leq p} = r^{\geq q-K, \leq q}$, we have $r'^{\geq p-K} = r^{\geq q-K}$. Then, for $n \geq p - K$,

$$D\mathcal{U}_I E \in r'(n) \text{ iff } r'^{\geq n} \text{ realizes } D\mathcal{U}_I E.$$

From now on assume $n < p - K$. If $I = [c_1, c_2]$ then, since $c_1, c_2 \in [0, K]$, we cannot exceed p. Then, **(R3)** holds.

Now, consider $I = [c_1, \infty)$, where $c_1 \in [0, K]$. As $r'^{\geq p-K} = r^{\geq q-K}$, if already $r'^{\geq n, \leq p}$ realizes $D\mathcal{U}_I E$, then $D\mathcal{U}_I E \in r'(n)$. Otherwise, assume that $r'^{\geq n, \leq p}$ does not realize $D\mathcal{U}_I E$. Then, for $n < p - K$, we have that $D\mathcal{U}_I E \in r(n)$ iff $D \in r(n), \ldots, r(p)$ and $D\mathcal{U} E \in r(p) = r'(p)$.

As $r'^{\leq p} = r^{\leq p}$ and $r'(p) = r(q)$, we have $D\mathcal{U}_I E \in r'(n)$ iff $D \in r'(n), \ldots, r'(p)$ and $r'^{\geq p}$ realizes $D\mathcal{U} E$. Then, for $n < p - K$, $D\mathcal{U}_I E \in r'(n)$ iff $r'^{\geq n}$ realizes $D\mathcal{U}_I E$. That is, **(R3)** holds.

This finishes the proof of the Claim.

An *n-sequence* of types is just a finite sequence of types $s(0) \cdots s(n-1)$. For $k \in \mathbb{N} \cup \{\infty\}$, we say that an n-sequence $s(0) \cdots s(n-1)$ *appears k times in r* if there are k distinct $j \geq 0$ such that $r(j + l) = s(l)$, for all $0 \leq l < n$.

Let r be a run through S with $r(i) = t$ and choose $k_1 \geq i$ minimal such that every K-sequence appearing in $r^{\geq k_1}$ appears infinitely often. We argue that it is without loss of generality that, between i and k_1, every K-sequence appears at most once: by the Claim, we can cut sequences that appear more than once. As there are at most $(\sharp_{C,\mathcal{T}})^K$ such K-sequences, we can assume that $k_1 \leq i + (\sharp_{C,\mathcal{T}})^K$.

Now, let U be the set of $C\mathcal{U}_I D \in r(k_1)$ that are not realized in $r^{\geq k_1, \leq k_1 + K}$, and choose $n \geq k_1$ minimal such that $r^{\geq k_1, \leq k_1+K} = r^{\geq n, \leq n+K}$ and each $C\mathcal{U}_I D \in U$ is realized in $r^{\geq k_1, <n}$. Let m_1 be minimal such that $r^{\geq k_1, \leq m_1}$ realizes some $D\mathcal{U}_I E \in U$. Reasoning as above using the Claim yields that we can assume without loss of generality that $m_1 \leq k_1 + (\sharp_{C,\mathcal{T}})^K$. Let now $m_2 > m_1$ be minimal such that $r^{\geq k_1, \leq m_2}$ realizes *two* $C\mathcal{U}_I D \in U$. As before, we can show that without loss of generality $m_2 \leq k_1 + 2(\sharp_{C,\mathcal{T}})^K$. Continuing this reasoning, we can conclude that, without loss of generality, $n \leq k_1 + |\mathsf{cl}(C,\mathcal{T})| \cdot (\sharp_{C,\mathcal{T}})^K$.

It is now routine to verify that $r' = r^{<k_1} * (r^{\geq k_1, <k_2})^{\omega}$, with $k_2 = n - k_1$, is a run through S. ❑

We are now in a position to show that the above procedure can be implemented using only (non-deterministic) exponential space. Obviously, the sets $S_0, \ldots, S_m$ and the numbers $n_0, \ldots, n_{m-1}$ can be stored in exponential space. Moreover, steps 2(a) and 2(b) can clearly be checked in exponential space. For 2(c), Lemma 3 implies that at most $\sharp_{C,\mathcal{T}} \cdot \sharp_{C,\mathcal{T}} \cdot \ell_1$, that is, double-exponentially many, pairs (t, i) have to be considered, but only one at a time. Finally, (the proof of) Lemma 4 enables the following algorithm for checking the existence of a run. First, guess binary numbers k_1, k_2 as in Lemma 4; then, guess a run in the form of the lemma. For the latter, proceed in a "sliding window" fashion: keep K consecutive types and verify **(R1)**-**(R3)** for the first type in the sequence, then drop that type, guess the next type, and continue. For detecting the loop, store the sequence $r(k_1), \ldots, r(k_1 + K)$ and verify that it appears again at $r(k_1 + k_2), \ldots, r(k_1 + k_2 + K)$ and, moreover, that each $D\mathcal{U}_I E \in r(k_1)$ is realized before k_2. We conclude the desired result.

Theorem 1. *Satisfiability in $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ is* EXPSPACE*-complete.*

3.2 $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$

In this section, we consider $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$, a well-behaved, yet expressive, fragment of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ in which intervals can only be of the form $[0, c]$ or $[c, \infty)$. This sort of intervals is useful to set maximum (deadline) points and minimum (initial) ones. For example, the CI

$$\mathsf{PhDStudent} \sqcap \exists \mathsf{defends}.\mathsf{Thesis} \sqsubseteq \Diamond_{[0,4]} \exists \mathsf{submits}.\mathsf{RevisedThesis}$$

says that 'PhD students who defend their thesis must submit a revised version within 4 weeks '.

We show that allowing intervals of this restricted form does not increase the complexity of satisfiability compared to $\mathcal{ALC}$ or LTL$_{\mathcal{ALC}}$, for both of which concept satisfiability is ExpTime-complete [25]. We concentrate again on the upper bound since ExpTime-hardness follows from satisfiability in $\mathcal{ALC}$ [12].

Our algorithm relies again on quasimodels; however, we will slightly adapt the definition of types to address the restricted intervals. As a consequence, it will suffice to consider quasimodels of the form (1) where n is only *single-exponentially* bounded, finally yielding an ExpTime decision procedure.

We first adapt the notion of a type. Instead of (‡), we define $\mathsf{cl}(C,\mathcal{T})$ as the closure under single negations of $\mathsf{sub}(C,\mathcal{T})$ extended with

$$\{D\mathcal{U}_{[0,c]}E, D\mathcal{U}_{[c,\infty)}E \mid c \in [0,K], D\mathcal{U}_I E \in \mathsf{sub}(C,\mathcal{T})\}.$$

Based on this, it is straightforward to show that Lemma 2 remains true for LTL$^{0,\infty}_{\mathcal{ALC}}$.

Note that there are now double-exponentially many types which typically prohibits an ExpTime decision procedure based on type elimination [28]. However, it is easy to see that a type t appearing in some quasimodel satisfies the following property.

(P) For every $D\mathcal{U}_{I_1}E, D\mathcal{U}_{I_2}E \in \mathsf{cl}(C,\mathcal{T})$ with $I_1 \subseteq I_2$, we have $D\mathcal{U}_{I_1}E \in t$ implies $D\mathcal{U}_{I_2}E \in t$.

To see **(P)**, fix some quasimodel $(S,\mathfrak{R})$ for C and $\mathcal{T}$ and assume that $D\mathcal{U}_{I_1}E \in t$ for some $t \in S(n)$, $n \geq 0$. By Condition **(Q2)**, there is a run $r \in \mathfrak{R}$ such that $r(n) = t$. As $D\mathcal{U}_{I_1}E \in t$, by Condition **(R3)**, $r^{\geq n}$ realizes $D\mathcal{U}_{I_1}E$. Since $I_1 \subseteq I_2$, $r^{\geq n}$ also realizes $D\mathcal{U}_{I_2}E$. By Condition **(R3)** again, $D\mathcal{U}_{I_2}E \in r(n)$.

Thus, it suffices to consider only types that satisfy **(P)**, whose number $\sharp_{C,\mathcal{T}}$ is bounded by $(2\cdot K)^{2|\mathsf{sub}(C,\mathcal{T})|}$, that is, exponential. From now on assume w.l.o.g. that $\mathsf{tp}(C,\mathcal{T})$ is the set of types in which **(P)** holds. The next lemma shows that we can assume that our quasimodels reach a periodic quasistate after at most exponentially many quasistates.

Lemma 5. *There is a quasimodel for C and $\mathcal{T}$ iff there is a quasimodel $(S,\mathfrak{R})$ for C and $\mathcal{T}$ of the form*

$$S = S_0 \dots S_{n-1}(S_n)^\omega,$$

for quasistates $S_0,\dots,S_n$ with $S_i \supsetneq S_{i+1}$, $0 \leq i < n \leq \sharp_{C,\mathcal{T}}$.

The proof proceeds in two steps, as in Lemma 2. In the first step we modify our quasimodel so that each quasistate at time $i+1$ is contained in the quasistate at time i. But now, in the second step, we show that if *two* consecutive quasistates coincide, that is, $S(i) = S(i+1)$ for some $i \geq 0$, then we can assume that all subsequent quasistates coincide as well, that is, $S(j) = S(i)$ for all $j \geq i$.

Based on Lemma 5, we now present an algorithm that performs type elimination, similar to what has been done for LTL$_{\mathcal{ALC}}$ [25].

Define $\rho(n) = \mathsf{min}\{\sharp_{C,\mathcal{T}}, n\}$, for all $n \geq 0$, and, moreover, define an operation "-1" on intervals as follows: $[0,c]-1 = [0,c-1]$ and $[c+1,\infty)-1 = [c,\infty)$, for all $c \geq 0$, and $[0,\infty)-1 = [0,\infty)$. We say that types t and t' are *compatible* if the following holds:

- $\bigcirc D \in t$ iff $D \in t'$, for all $\bigcirc D \in \mathsf{cl}(C,\mathcal{T})$; and
- $D\mathcal{U}_I E \in t$ iff either (the sequence) $t\,t'$ realizes $D\mathcal{U}_I E$, or $\{D, D\mathcal{U}_{I-1}E\} \subseteq t'$, for all $D\mathcal{U}_I E \in \mathsf{cl}(C,\mathcal{T})$.

The algorithm starts with sets

$$S_0,\dots,S_{n-1},S_n$$

where $n = \sharp_{C,\mathcal{T}}$ and each S_i is initially set to $\mathsf{tp}(C,\mathcal{T})$. We then exhaustively eliminate types t from some S_i, $0 \leq i \leq n$ if t violates one of the following conditions:

(T1) for all $\exists r.D \in t$, there is $t' \in S_i$ such that $\{D\} \cup \{\neg E \mid \neg\exists r.E \in t\} \subseteq t'$;
(T2) there is $t' \in S_{\rho(i+1)}$ such that t and t' are compatible;
(T3) if $i > 0$, there is $t' \in S_{i-1}$ such that t' and t are compatible;
(T4) for all $D\mathcal{U}_I E \in t$, there is $k \geq 0$ and a sequence

$$t_1 \in S_{\rho(i+1)},\dots,t_k \in S_{\rho(i+k)}$$

such that $t_0 \cdots t_k$ (with $t_0 = t$) realizes $D\mathcal{U}_I E$, and t_l and t_{l+1} are compatible, for all $0 \leq l < k$.

Before giving details on how to implement the conditions, especially **(T4)**, we finish the description of the algorithm and show correctness. The algorithm stops when no further types can be eliminated. It returns 'satisfiable' if there is a surviving $t \in S_0$ with $C \in t$, and 'unsatisfiable', otherwise.

Lemma 6. *The algorithm returns 'satisfiable' iff there is a quasimodel for C and $\mathcal{T}$.*

Proof. For ($\Rightarrow$), let $S^*_0,\dots,S^*_n$ be the result of the type elimination procedure. Define $(S^*,\mathfrak{R})$ with $S^* = S^*_0 \dots S^*_{n-1}(S^*_n)^\omega$ and $\mathfrak{R}$ as the set of all sequences r of types such that, for all $i \geq 0$:

1. $r(i) \in S^*(i)$;
2. $r(i)$ and $r(i+1)$ are compatible; and
3. $D\mathcal{U}_I E \in r(i)$ iff $r^{\geq i}$ realizes $D\mathcal{U}_I E$, for all $D\mathcal{U}_I E \in \mathsf{cl}(C,\mathcal{T})$.

We now argue that $(S^*,\mathfrak{R})$ is a quasimodel. By **(T1)**, the sets $S^*_0,\dots,S^*_n$ generated by the algorithm are quasistates, so S^* is a sequence of quasistates. By assumption, there is $t \in S^*_0$ with $C \in t$, which gives us **(Q1)**. By definition of $\mathfrak{R}$, we have that every $r \in \mathfrak{R}$ is a run through S^* (see **(R1)**-**(R3)**). Then, for **(Q2)**, we only need to see that for every $t \in S^*(j)$ there is $r \in \mathfrak{R}$ such that $r(j) = t$, $j \in \mathbb{N}$. Let $r' = r^{\leq j}$ be a sequence of types such that all consecutive types in r' are compatible and $r'(j) = t$. By **(T3)** such sequence exists. We now extend this run using **(T2)** and **(T4)**. Assume that there is no $D\mathcal{U}_I E \in r'(j)$. Then by **(T2)** there is $t' \in S^*(j+1)$ such that t and t' are compatible. So we extend r' with t'. Now assume there is $D\mathcal{U}_I E \in r'(j)$. By **(T4)** there is a minimal sequence $t_0 t_1 \cdots t_k$ of types that realizes all $D\mathcal{U}_I E \in r'(j)$. We extend r' with t_1. Continuing with this process one can ensure the existence of an infinite sequence satisfying all conditions of the sequences in $\mathfrak{R}$.

For the other direction ($\Leftarrow$), assume there is a quasimodel, which is without loss of generality of the form $S' = S'_0 \dots S'_{n-1}(S'_n)^\omega$, by Lemma 5. Let $S^*_0,\dots,S^*_n$ be the result of the type elimination. It is routine to verify that $S'_i \subseteq S^*_i$, $0 \leq i \leq n$ by showing that no type in S'_i violates **(T1)**-**(T4)**. Clearly, each type satisfies **(T1)**, since $(S',\mathfrak{R})$ is a quasimodel. Moreover, conditions **(T2)**-**(T4)** are consequences of the existence of runs through each type.

Observe finally that, by **(Q1)**, there is some $t \in S'_0$, thus $t \in S^*_0$, with $C \in t$, that is, the algorithm returns 'satisfiable'. ❏

It is not hard to see that the algorithm runs in exponential time. The maintained sets of types have initally exponential size and in every step some type is eliminated. Conditions **(T1)**-**(T3)** can clearly be checked in exponential time. Finally, **(T4)** can be cast as a reachability problem, which can be solved in polynomial time, in the following (exponentially sized) graph: vertices (t,i) for all $t \in S_i$ and edges between (t,i) and $(t',\rho(i+1))$ iff t and t' compatible. We thus conclude:

Theorem 2. *Satisfiability in* $\mathrm{LTL}^{0,\infty}_{\mathcal{ALC}}$ *is* ExpTime*-complete.*

4 $\mathrm{MTL}_{\mathcal{ALC}}$

In this section, we investigate a TDL that emerges from combining the real-time logic MTL (over the naturals) and $\mathcal{ALC}$. $\mathrm{MTL}_{\mathcal{ALC}}$*-concepts* are formed according to the following rule

$$C, D ::= A \mid \neg C \mid C \sqcap D \mid \exists r.C \mid \bigcirc_I C \mid C \mathcal{U}_I D,$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and I is an interval.

Note that $\mathrm{MTL}_{\mathcal{ALC}}$-concepts are formed just like $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$-concepts except for the constructor $\bigcirc_I$. The main difference between $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$ and $\mathrm{MTL}_{\mathcal{ALC}}$ lies in their semantics: $\mathrm{MTL}_{\mathcal{ALC}}$ is a timed extension of $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$, in other words, each interpretation in $(\mathcal{I}_n)_{n\in\mathbb{N}}$ explicitly refers to its time (think of it as the reading of a'fictitious discrete clock') allowing to quantitatively reason about time delays.

Formally, a *timed interpretation* $\mathfrak{I}$ is a tuple $(\Delta^{\mathfrak{I}}, (\mathcal{I}_n)_{n\in\mathbb{N}}, \tau)$ with $\tau : \mathbb{N} \to \mathbb{N}$ a mapping with $\tau(n) < \tau(n+1)$, for all $n \in \mathbb{N}$, which specifies that the n-th interpretation happens to be at time point $\tau(n)$. Note that there might be *gaps* between two interpretations, e.g., when $\tau(3) = 8$ and $\tau(4) = 10$, then there is no interpretation at time point 9. Intuitively, we view $(\mathcal{I}_n, \tau(n))_{n\geq 0}$ as a *sequence of observations*, for instance, in a real-time system, and then understand the difference $\tau(n+1) - \tau(n)$ as the time delay between observations n and $n+1$.

The interpretation function $\cdot^{\mathfrak{I},n}$ is lifted to complex concepts as in Section 2 for the constructors $\neg$, $\sqcap$, and $\exists r.C$. For $\bigcirc_I$ and $\mathcal{U}_I$, it is defined as follows:

$$\begin{aligned}(\bigcirc_I C)^{\mathfrak{I},n} &= \{d \mid d \in C^{\mathfrak{I},n+1} \wedge \tau(n+1) - \tau(n) \in I\},\\ (C\mathcal{U}_I D)^{\mathfrak{I},n} &= \{d \mid \exists k > n : d \in D^{\mathfrak{I},k} \wedge \tau(k) - \tau(n) \in I\\ &\qquad \wedge \forall m \in (n,k) : d \in C^{\mathfrak{I},m}\}.\end{aligned}$$

One could expect that, just like for $\mathrm{LTL}^{\mathsf{bin}}_{\mathcal{ALC}}$, the complexity of satisfiability in $\mathrm{MTL}_{\mathcal{ALC}}$ is not higher than in the components; in particular, ExpSpace-complete as in MTL [1]. Surprisingly, we prove that there is an exponential jump in the complexity; the main reason for such an increase is that, due to slightly different semantics, the independence of elements in each $\mathcal{I}_n$ is lost.

Theorem 3. *Satisfiability in* $\mathrm{MTL}_{\mathcal{ALC}}$ *is* 2ExpSpace*-complete.*

We prove here only the lower bound. The upper bound will follow from a more general result, see Theorem 6 in Section 5.

Proof. We reduce the word problem of a double-exponentially space-bounded deterministic Turing machine. Fix that TM $\mathfrak{A} = (Q, \Sigma, \Gamma, \delta, q_0, F)$ with $\delta : Q \times \Gamma \to Q \times \Gamma \times \{l, r\}$ and assume that $\mathfrak{A}$ is 2^{2^n}-space bounded on inputs of length n. Let $Q' = Q \cup \{\overline{q}\}$, $k = |\Gamma \times Q'| + 1$ and fix some bijection $\pi\colon [1, k-1] \to \Gamma \times Q'$. We are going to use the following symbols:

- Tape, to mark the tape cells;
- $A_{a,q}$, $a \in \Gamma, q \in Q'$, to label cells with a symbol a and a state q; $\overline{q}$ expresses that the head is somewhere else.

Recall that we use the abbreviations $\Diamond_i$ and $\bigcirc_i$ instead of $\Diamond_{[i,i]}$ and $\bigcirc_{[i,i]}$, and just $\Diamond$ instead of $\Diamond_{[0,\infty)}$. For inputs of length n, we will construct a TBox $\mathcal{T}_n$, whose basic ingredients are the following concept inclusions:

$$\mathsf{Tape} \sqsubseteq \Diamond_{[0,k]}\mathsf{Tape} \sqcap \Box_{[0,k-1]}\neg\mathsf{Tape} \tag{2}$$

$$\mathsf{Tape} \sqsubseteq \bigcirc_{[0,k-1]}\top \tag{3}$$

$$\bigcirc_i \top \equiv A_{\pi(i)}, \qquad \text{for all } i \in [1, k-1] \tag{4}$$

Intuitively, using CI (2), we enforce that every k-th time point is labeled with Tape. By CI (3), we express that, if Tape is observed, the next observation is due within 1 to $k-1$ time points, but there is a choice. Finally, using CI (4), we *globally* mark all domain elements in a world, depending on the delay of the next observation, with some $A_{a,q}$, that is, information about state and tape symbol.

It remains to show how to synchronize consecutive configurations. Basically, the technique goes back to the following well-known lemma [21, Lemma 3.3], which is based on [20, Lemma 4.1] itself.

Lemma 7. *For each* $n \geq 1$*, there is a satisfiable formula* φ_n *in propositional temporal logic extended with* $\bigcirc^n$*,* n *in binary, of size* $O(n)$*, and some* $M \geq 0$ *such that* $\varphi_n \models \bigcirc^m p_2$ *iff* $m = M + j \cdot 2^n \cdot 2^{2^n}$*, for some* $j \geq 0$*.*

Using (the proof of) this well-known result, one can define a concept C_n that satisfies an analogous property, namely

$$\mathcal{T}_n \models C_n \sqsubseteq \Diamond_m P_2 \quad \text{iff} \quad m = j \cdot k \cdot 2^n \cdot 2^{2^n}, \text{for some } j.$$

We use this concept C_n (without giving details on the shape of C_n) to describe the remaining relevant parts in $\mathcal{T}_n$. We include the following concept inclusions:

- For $a \in \Gamma$ and $\overline{q}$:

$$\mathsf{Tape} \sqcap A_{a,\overline{q}} \sqsubseteq \exists r.(C_n \sqcap \neg P_2 \mathcal{U}(P_2 \sqcap \bigsqcup_{q' \in Q'} A_{a,q'})) \tag{5}$$

- For $a \in \Gamma$, $q \in Q$, and $\delta(q,a) = (\_, b, \_)$:

$$\mathsf{Tape} \sqcap A_{a,q} \sqsubseteq \exists r.(C_n \sqcap \neg P_2 \mathcal{U}(P_2 \sqcap A_{b,\overline{q}})) \tag{6}$$

- For $a \in \Gamma$, $q \in Q$, and $\delta(q,a) = (q', \_, r)$:

$$\mathsf{Tape} \sqcap A_{a,q} \sqsubseteq \Diamond_k \exists r.(C_n \sqcap \neg P_2 \mathcal{U}(P_2 \sqcap \bigsqcup_{b \in \Gamma} A_{b,q'})) \tag{7}$$

- For $a \in \Gamma$, $q \in Q$, and $\delta(q,a) = (q', \_, l)$:

$$\mathsf{Tape} \sqcap \Diamond_k A_{a,q} \sqsubseteq \exists r.(C_n \sqcap \neg P_2 \mathcal{U}(P_2 \sqcap \bigsqcup_{b \in \Gamma} A_{b,q'})) \tag{8}$$

Let $N = k \cdot 2^n \cdot 2^{2^n}$. Intuitively, CI (5) states that a world labeled with $\overline{q}$ is labeled with the same symbol in the next configuration, that is, N tape cells later. CI (6) ensures that, if a world is labeled with (a, q), then the corresponding world N tape cells later is labeled with b when $\delta(q,a) = (\_, b, \_)$; the corresponding state is $\overline{q}$ as the head moves left or right. Finally, CIs (7) and (8) make sure that the head is moved according to the transition. It remains to ensure that the non-head worlds are labeled with $\overline{q}$. For this, one has to take into account the environment of a cell, as illustrated by the following CI:

$$\mathsf{Tape} \sqcap A_{a_1,q_1} \sqcap \Diamond_k(\neg P_2 \sqcap A_{a_2,q_2} \sqcap \Diamond_k(\neg P_2 \sqcap A_{a_3,q_3}))) \sqsubseteq \Diamond_k \exists r.(C_n \sqcap \neg P_2 \mathcal{U}(P_2 \sqcap \bigsqcup_{b \in \Gamma} A_{b,\overline{q}})),$$

if $q_1 = q_2 = q_3 = \overline{q}$ or $\delta(q_1, a_1) = (\_, \_, l)$ or $\delta(q_3, a_3) = (\_, \_, r)$.

The remaining cases are similar. In particular, at cells close to the left or right border of a configuration, it suffices to take a smaller environment into account.

Now, let $w = a_1 \cdots a_n$ be some input word for $\mathfrak{A}$. Define a concept C_w by taking:

$$C_w = \mathsf{Tape} \sqcap C_n \sqcap A_{a_1,q_0} \sqcap \prod_{i=1}^{n-1} \Diamond_{ik} A_{a_{i+1},\overline{q}} \sqcap \Diamond_{(n-1)k}(A_{\not\vartriangleright,\overline{q}} \mathcal{U} P_2) \sqcap \Diamond \bigsqcup_{a \in \Gamma, q \in F} A_{a,q}.$$

Intuitively, Tape ensures a computation is initiated, C_n ensures that the tape is separated into configurations, A_{a_1,q_0} and the big conjunction enforces that the input word is written on the tape and $A_{b,\overline{q}}\,\mathcal{U}\,P_2$ ensures that the remaining cells are labeled with blank b and are non-head states. Finally, the last conjunct expresses that a final state is reachable. Based on the construction, it is not hard to verify the following claim, which finishes the reduction.

Claim. $\mathfrak{A}$ accepts w of length n if there is a model of C_w and $\mathcal{T}_n$. ❑

4.1 $\text{MTL}^{0,\infty}_{\mathcal{ALC}}$

Restricting the intervals to the form $[0, c]$ and $[c, \infty)$ leads to better complexity also for $\text{MTL}_{\mathcal{ALC}}$; however, not to ExpTime as for $\text{LTL}^{0,\infty}_{\mathcal{ALC}}$. To see this, we sketch here how to adapt the reduction used in the previous theorem to get an ExpSpace-lower bound. A matching upper bound follows from Theorem 8 below.

Recall that CIs (2)-(4) provide the central idea of the reduction. While (2) and (3) are already in $\text{MTL}^{0,\infty}_{\mathcal{ALC}}$, we replace (4) with CIs

$$\neg\Diamond_{[0,i]}\top \sqcap \prod_{l=i+2}^{k-1} \neg A_{\pi(l)} \sqsubseteq A_{\pi(i+1)}, \quad \text{and} \tag{9}$$

$$A_{\pi(i)} \sqsubseteq \neg\Diamond_{[0,i-1]}\top, \tag{10}$$

for all $0 \leq i < k-1$. Intuitively, (9) expresses that if there is a gap of at least i (realized by $\neg\Diamond_{[0,i]}\top$) and all $A_{\pi(l)}$ for $l > i+1$ are not satisfied, that is, there is no larger gap, conclude $A_{\pi(i+1)}$. Together with (10), this implies that, again, a unique $A_{a,q}$ is satisfied for *all* domain elements in a world. Note that, as there is a fixed Turing machine with an ExpSpace-hard word problem, k is fixed and we do not require succinct encoding here.

The remainder of the above proof deals with synchronizing information between consecutive configurations. While the concept C_n can certainly not be defined in $\text{MTL}^{0,\infty}_{\mathcal{ALC}}$, we can use the succinct intervals to communicate between tape cells that are *exponentially* far away. For instance, we can mark every $N = k \cdot 2^n$-th time point with a concept name X by $\mathsf{X} \sqsubseteq \Diamond_{[0,N]}\mathsf{X} \sqcap \Box_{[0,N-1]}\neg\mathsf{X}$. We thus get:

Theorem 4. *Satisfiability in* $\text{MTL}^{0,\infty}_{\mathcal{ALC}}$ *is* ExpSpace*-complete.*

5 Temporal TBoxes

We now take a look at the case where temporal operators can also be applied to concept inclusions in the TBox, which adds means for expressing dynamics of global information, e.g., in norms.

5.1 Temporal TBoxes in $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$

Temporal $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$*-TBoxes* are defined by the following grammar:

$$\varphi, \psi ::= C \sqsubseteq D \mid \neg\varphi \mid \varphi \wedge \psi \mid \bigcirc\varphi \mid \varphi\mathcal{U}_I\psi,$$

where C, D are $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-concepts, I an interval. We define the truth relation $\mathfrak{I}, n \models \varphi$ (with $\mathfrak{I}$ an interpretation and $n \in \mathbb{N}$ a time point) by starting with $\mathfrak{I}, n \models C \sqsubseteq D$ iff $C^{\mathfrak{I},n} \subseteq D^{\mathfrak{I},n}$, and extending it to the complex TBox formulas analogously to Section 2; e.g., $\mathfrak{I}, n \models \bigcirc\varphi$ iff $\mathfrak{I}, n+1 \models \varphi$. $\mathfrak{I}$ is a *model* of a temporal $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-TBox φ if $\mathfrak{I}, 0 \models \varphi$.

We are concerned with the problem of *temporal TBox satisfiability*, that is, the problem of deciding whether a given temporal TBox φ has a model. Note that, in contrast to Section 3, a concept is not part of the input because there is a model of a concept C and a temporal TBox φ if and only if the temporal TBox $\neg(\top \sqsubseteq \neg C) \wedge \varphi$ is satisfiable.

Temporal TBoxes are useful to set the dynamics of protocols or norms. For example, the temporal $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-TBox

$$\Diamond_{[4,4]}(\mathsf{PhDStud} \sqcap \exists\mathsf{defends}.\mathsf{Thesis} \sqcap \neg\exists\mathsf{has}.\mathsf{ConfPub} \sqsubseteq \bigcirc\exists\mathsf{takes}.\mathsf{ValidationExam})$$

says that after 4 years there will be a norm stating that all PhD students who defend their thesis and do not have a conference publication will need to take a validation exam the year after that.

The first result here is that the complexity of temporal TBox satisfiability is exponentially higher than for concept satisfiability relative to global TBoxes; notably, the lower bound is a consequence of Theorem 3 above.

Theorem 5. *Satisfiability of temporal* $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$*-TBoxes is* 2ExpSpace*-complete.*

Membership in 2ExpSpace is a consequence of the following: (i) satisfiability of temporal $\text{LTL}_{\mathcal{ALC}}$-TBoxes is ExpSpace-complete [25], and (ii) any $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$ temporal TBox can be translated into an equivalent though exponentially larger $\text{LTL}_{\mathcal{ALC}}$-TBox, by expanding the succinctly encoded intervals.

For the lower bound, we reduce the satisfiability problem for $\text{MTL}_{\mathcal{ALC}}$, which is 2ExpSpace-hard cf. Theorem 3. Introduce a fresh concept name Gap, which intuitively models the "gaps" between consecutive observations in $\text{MTL}_{\mathcal{ALC}}$, and define the map $\cdot^\dagger$ inductively by taking:

$$\begin{aligned}
A^\dagger &= A \\
(\neg C)^\dagger &= \neg(C^\dagger) \\
(C \sqcap D)^\dagger &= C^\dagger \sqcap D^\dagger \\
(\exists r.C)^\dagger &= \exists r.C^\dagger \\
(\bigcirc_I C)^\dagger &= \mathsf{Gap}\,\mathcal{U}_I(\neg\mathsf{Gap} \sqcap C^\dagger) \\
(C\,\mathcal{U}_I\,D)^\dagger &= (\mathsf{Gap} \sqcup C^\dagger)\,\mathcal{U}_I(\neg\mathsf{Gap} \sqcap D^\dagger) \\
(C \sqsubseteq D)^\dagger &= (\neg\mathsf{Gap} \sqcap C^\dagger \sqsubseteq D^\dagger)
\end{aligned}$$

It is routine to verify that:

Lemma 8. *An* $\text{MTL}_{\mathcal{ALC}}$*-concept* C *and TBox* $\mathcal{T}$ *are satisfiable iff the following temporal* $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$*-TBox is satisfiable:*

$$\neg(\top \sqsubseteq \mathsf{Gap} \sqcup \neg C^\dagger) \wedge \Box \bigwedge_{\alpha \in \mathcal{T}} \alpha^\dagger \wedge \Box(\top \sqsubseteq \mathsf{Gap} \vee \top \sqsubseteq \neg\mathsf{Gap}) \wedge \Box\Diamond(\top \sqsubseteq \neg\mathsf{Gap}).$$

5.2 Temporal TBoxes in $\text{MTL}_{\mathcal{ALC}}$

The syntax of *temporal* $\text{MTL}_{\mathcal{ALC}}$*-TBoxes* is obtained from the syntax of $\text{LTL}^{\text{bin}}_{\mathcal{ALC}}$-TBoxes by just replacing $\bigcirc\varphi$ with $\bigcirc_I\varphi$. The semantics is adapted accordingly as discussed in Section 4; for instance

$$\mathfrak{I}, n \models \bigcirc_I\varphi \quad \text{iff} \quad \mathfrak{I}, n+1 \models \varphi \text{ and } \tau(n+1) - \tau(n) \in I.$$

Theorem 6. *Satisfiability of temporal* $\text{MTL}_{\mathcal{ALC}}$*-TBoxes is* 2ExpSpace*-complete.*

The lower bound is inherited from Theorem 3. For the upper bound, we lift the mapping $\cdot^\dagger$ given in the proof of Theorem 5 to temporal MTL$_{\mathcal{ALC}}$-TBoxes. For a temporal MTL$_{\mathcal{ALC}}$-TBox φ, define $\varphi^\dagger$ inductively as follows:

$$\begin{aligned}(\varphi \wedge \psi)^\dagger &= \varphi^\dagger \wedge \psi^\dagger\\ (\neg\varphi)^\dagger &= \neg(\varphi^\dagger)\\ (\bigcirc_I \varphi)^\dagger &= (\top \sqsubseteq \mathsf{Gap})\,\mathcal{U}_I(\top \sqsubseteq \neg\mathsf{Gap} \wedge \varphi^\dagger)\\ (\varphi\,\mathcal{U}_I\psi)^\dagger &= (\top \sqsubseteq \mathsf{Gap} \vee \varphi^\dagger)\,\mathcal{U}_I(\top \sqsubseteq \neg\mathsf{Gap} \wedge \psi^\dagger)\end{aligned}$$

The following Lemma, which is proved similar to Lemma 8, together with the fact that satisfiability of temporal LTL$^{\mathsf{bin}}_{\mathcal{ALC}}$-TBoxes can be checked in 2ExpSpace concludes the upper bound.

Lemma 9. *A temporal* MTL$_{\mathcal{ALC}}$*-TBox* φ *is satisfiable iff* $\varphi^\dagger \wedge \Box(\top \sqsubseteq \mathsf{Gap} \vee \top \sqsubseteq \neg\mathsf{Gap}) \wedge \Box\Diamond(\top \sqsubseteq \neg\mathsf{Gap})$ *is satisfiable.*

5.3 Restriction to intervals $[0, c], [c, \infty)$

We have seen in Theorem 2 that the restriction to intervals of the form $[0, c], [c, \infty)$ leads to better complexity in the case of (classical) satisfiability. We show here that this in fact also applies to temporal TBoxes. In fact, the observations made in Section 3.2 apply here as well and it is fairly straightforward to extend it to this more general setting. The upper bound is then obtained by adapting a strategy that has been used for monodic first-order temporal logic, $\mathcal{QTL}_{\mathcal{U}\Box}$ in [15, Theorem 11.30].

Due to this proximity, we sketch only the necessary changes. We need to extend the definition of a type to reflect the information about the TBox formulas as follows. For a TBox formula φ, denote with $\mathsf{sub}(\varphi)$ the set of all subformulas of φ together with all subconcepts appearing in some of these subformulas; in particular, $\mathsf{sub}(\varphi)$ can contain both a concept inclusion $C \sqsubseteq D$ and a concept C. Similar to Section 3.2, $\mathsf{cl}(\varphi)$ is the closure under single negation of $\mathsf{sub}(\varphi)$ extended with the set

$$\{\alpha\,\mathcal{U}_{[0,c]}\beta, \alpha\,\mathcal{U}_{[c,\infty)}\beta \mid c \in [0, K], \alpha\,\mathcal{U}_I\beta \in \mathsf{sub}(\varphi)\},$$

where K is the largest constant in φ and α, β could be concepts or TBox formulae. Now, a type is a subset $t \subseteq \mathsf{cl}(\varphi)$ such that:

- $\alpha \in t$ iff $\neg\alpha \notin t$, for all $\neg\alpha \in \mathsf{cl}(\varphi)$;
- $\psi \wedge \chi \in t$ iff $\{\psi, \chi\} \subseteq t$, for all $\psi \wedge \chi \in \mathsf{cl}(\varphi)$;
- $D \sqcap E \in t$ iff $\{D, E\} \subseteq t$, for all $D \sqcap E \in \mathsf{cl}(\varphi)$;
- $C \sqsubseteq D \in t$ and $C \in t$ implies $D \in t$.

As argued in Section 3.2, we only need to consider those (exponentially many) types, which satisfy property **(P)**, appropriately lifted to include TBox formulas. A *quasistate for* φ is a set of types with the additional requirement that the types *agree on the TBox formulas*, that is, $\psi \in t$ iff $\psi \in t'$ for types t, t' in the same quasistate, and all TBox subformulas ψ. After lifting also the run condition **(R3)** to apply to TBox formulas, the notion of a quasimodel remains (almost) identical: a *quasimodel for* φ is a pair $(S, \mathfrak{R})$ such that:

(Q1) $\varphi \in t$ for some $t \in S(0)$; and
(Q2) for all $t \in S(n)$, $n \geq 0$ there is a run $r \in \mathfrak{R}$ such that $r(n) = t$.

As before, the existence of a quasimodel for φ characterizes satisfiability of φ. Moreover, if there is a quasimodel, then there is a quasimodel $(S, \mathfrak{R})$ of the regular form

$$S = S(0) \dots S(n-1)(S(n) \dots S(n+m-1))^\omega$$

with $n \leq \sharp\mathsf{qs}_\varphi$ and $m \leq |\mathsf{sub}(\varphi)| \cdot \sharp\mathsf{qs}_\varphi \cdot (\sharp_\varphi)^2 + \sharp\mathsf{qs}_\varphi$, where $\sharp_\varphi$ and $\sharp\mathsf{qs}_\varphi$ denote the number of types and quasistates for φ, respectively. Thus, both the length n of the initial part and the length m of the cycle are double exponentially bounded.

Based on this, one can devise the following algorithm, similar to [15, Lemma 11.30] and the algorithm for the proof of Theorem 1: guess numbers n, m within the mentioned bounds, and step by step the sequence S keeping always only two consecutive quasistates. While guessing the sequence, verify on the fly that each type in $S(i)$ has a compatible type in $S(i+1)$, and vice versa. At time point n, store $S(n)$ and continue m more steps until reaching $S(m+n) = S(n)$. Moreover, verify that all $\alpha\,\mathcal{U}_I\beta$ appearing in some type in $S(n)$ are realized on the way to $S(n+m)$. It should be clear that this can be done in (non-deterministic) exponential-space, yielding:

Theorem 7. *Satisfiability of temporal* LTL$^{0,\infty}_{\mathcal{ALC}}$*-TBoxes is* ExpSpace*-complete.*

As a consequence of Lemma 9, we additionally obtain:

Theorem 8. *Satisfiability of temporal* MTL$^{0,\infty}_{\mathcal{ALC}}$*-TBoxes is* ExpSpace*-complete.*

6 Conclusions and Future Work

In this paper, we have launched the study of *metric* TDLs allowing for quantitative temporal reasoning, and established a fairly complete landscape of the complexity of satisfiability for LTL$^{\mathsf{bin}}_{\mathcal{ALC}}$ and MTL$_{\mathcal{ALC}}$ (over the naturals). Most interestingly, we have shown that the ability to reason explicitly about timestamps of observations brings additional computational complexity. In particular, the complexity of concept satisfiability is then the same as that of temporal TBox satisfiability, c.f. Table 1.

As immediate future work, we will investigate TDLs based on MTL with *continuous-semantics* (over the reals). For some applications, the continuous-semantics seems to be more appropriate in the sense that a real-time system is continuously observed instead of only when an event or action happens. The change from pointwise to continuous semantics is not for free since full MTL becomes undecidable; however, several decidable fragments have been already identified [27]. We plan to build on these results and study TDLs based on decidable fragments of MTL with continuous-semantics.

We will also look at quantitative TDLs in the context of *ontology-based data access (OBDA)* [14] over temporal databases. We believe that the present paper lays important foundations for understanding the *combined complexity* of the *query answering problem* with *m*TDLs. However, for *data complexity*, i.e., when only the data is considered as part of the input, TBoxes with succinctly represented intervals can be used for free. In this case, an interesting problem is to consider data timestamped with intervals, succinctly representing its validity time. In this scenario, it would be fruitful to study restrictions of *m*TDLs based on 'data-tractable' DLs such as those in *DL-Lite* [4] or $\mathcal{EL}$ [11], whose temporal extensions to access temporal (timestamped) data have been recently investigated [10, 6, 16, 22]. However, none of these works studies interval encoding of timestamps.

REFERENCES

[1] R. Alur and T. A. Henzinger, 'Real-time logics: Complexity and expressiveness', *Inf. Comput.*, **104**(1), 35–77, (1993).

[2] R. Alur and T. A. Henzinger, 'A really temporal logic', *J. ACM*, **41**(1), 181–204, (1994).

[3] A. Artale, D. Bresolin, A. Montanari, G. Sciavicco, and V. Ryzhikov, 'DL-Lite and interval temporal logics: a marriage proposal', in *Proc. of ECAI-14*, pp. 957–958, (2014).

[4] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev, 'The DL-Lite family and relations', *J. Artif. Intell. Res. (JAIR)*, **36**, 1–69, (2009).

[5] A. Artale and E. Franconi, 'A survey of temporal extensions of description logics', *Ann. Math. Artif. Intell.*, **30**(1-4), 171–210, (2000).

[6] A. Artale, R. Kontchakov, A. Kovtunova, V. Ryzhikov, F. Wolter, and M. Zakharyaschev, 'First-order rewritability of temporal ontology-mediated queries', in *Proc. of IJCAI-15*, pp. 2706–2712, (2015).

[7] A. Artale, R. Kontchakov, C. Lutz, F. Wolter, and M. Zakharyaschev, 'Temporalising tractable description logics', in *Proc. of TIME-07*, (2007).

[8] A. Artale, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev, 'A cookbook for temporal conceptual data modelling with description logics', *ACM Trans. Comput. Log.*, **15**(3), 25, (2014).

[9] A. Artale, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev, 'Tractable interval temporal propositional and description logics', in *Proc. of AAAI-15*, pp. 1417–1423, (2015).

[10] A. Artale, R. Kontchakov, F. Wolter, and M. Zakharyaschev, 'Temporal description logic for ontology-based data access', in *Proc. of IJCAI-13*, (2013).

[11] F. Baader, S. Brandt, and C. Lutz, 'Pushing the EL envelope', in *Proc. of IJCAI-05*, pp. 364–369, (2005).

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

[13] F. Baader, S. Ghilardi, and C. Lutz, 'LTL over description logic axioms', *ACM Trans. Comput. Log.*, **13**(3), 21, (2012).

[14] M. Bienvenu and M. Ortiz, 'Ontology-mediated query answering with data-tractable description logics', in *Proc. of Reasoning Web*, pp. 218–307, (2015).

[15] D. M. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev, *Many-dimensional modal logics: theory and applications*, volume 148, North Holland, 2003.

[16] V. Gutiérrez-Basulto, J. C. Jung, and R. Kontchakov, 'Temporalized $\mathcal{EL}$ ontologies for accessing temporal data: Complexity of atomic queries', in *Proc. of IJCAI-16*, (2016).

[17] V. Gutiérrez-Basulto, J. C. Jung, and C. Lutz, 'Complexity of branching temporal description logics', in *Proc. of ECAI-12*, (2012).

[18] V. Gutiérrez-Basulto, J. C. Jung, and T. Schneider, 'Lightweight description logics and branching time: a troublesome marriage', in *Proc. of KR-14*, (2014).

[19] V. Gutiérrez-Basulto, J.C. Jung, and T. Schneider, 'Lightweight temporal description logics with rigid roles and restricted TBoxes', in *Proc. of IJCAI-15*, pp. 3015–3021, (2015).

[20] J. Y. Halpern and M. Y. Vardi, 'The complexity of reasoning about knowledge and time. i. lower bounds', *J. Comput. Syst. Sci.*, **38**(1), 195–237, (1989).

[21] I. M. Hodkinson, R. Kontchakov, A. Kurucz, F. Wolter, and M. Zakharyaschev, 'On the computational complexity of decidable fragments of first-order linear temporal logics', in *Proc. of TIME-ICTL 2003*, pp. 91–98, (2003).

[22] R. Kontchakov, L. Pandolfo, L. Pulina, V. Ryzhikov, and M. Zakharyaschev, 'Temporal and spatial OBDA with many-dimensional Halpern-Shoham logic', in *Proc. of IJCAI-16*, (2016).

[23] R. Koymans, 'Specifying real-time properties with metric temporal logic', *Real-Time Systems*, **2**(4), 255–299, (1990).

[24] C. Lutz, D. Walther, and F. Wolter, 'Quantitative temporal logics over the reals: PSpace and below', *Inf. Comput.*, **205**(1), 99–123, (2007).

[25] C. Lutz, F. Wolter, and M. Zakharyaschev, 'Temporal description logics: A survey.', in *Proc. of TIME-07*, pp. 3–14, (2008).

[26] Z. Manna and P. Wolper, 'Synthesis of communicating processes from temporal logic specifications', *ACM Trans. Program. Lang. Syst.*, **6**(1), 68–93, (1984).

[27] J. Ouaknine and J. Worrell, 'Some recent results in metric temporal logic', in *Proc. of FORMATS-08*, pp. 1–13, (2008).

[28] V. Pratt, 'Models of program logics', in *Proc. of 20th Annual Symposium on Foundations of Computer Science*, pp. 115–122, (1979).

[29] K. Schild, 'Combining terminological logics with tense logic', in *Proc. of EPIA*, pp. 105–120. Springer, (1993).

[30] A. P. Sistla and E. M. Clarke, 'The complexity of propositional linear temporal logics', *J. ACM*, **32**(3), 733–749, (1985).

[31] F. Wolter and M. Zakharyaschev, 'Temporalizing description logics', in *Proc. of FroCoS-98*, pp. 379–402. Studies Press/Wiley, (1999).