A Cookbook for Temporal Conceptual Data Modelling with Description Logics

ALESSANDRO ARTALE, KRDB Research Centre, Free University of Bozen-Bolzano, Italy
ROMAN KONTCHAKOV, Department of Computer Science and Information Systems, Birkbeck, University of London, UK
VLADISLAV RYZHIKOV, KRDB Research Centre, Free University of Bozen-Bolzano, Italy
MICHAEL ZAKHARYASCHEV, Department of Computer Science and Information Systems, Birkbeck, University of London, UK

We design temporal description logics (TDLs) suitable for reasoning about temporal conceptual data models and investigate their computational complexity. Our formalisms are based on *DL-Lite* logics with three types of concept inclusions (ranging from atomic concept inclusions and disjointness to the full Booleans), as well as cardinality constraints and role inclusions. The logics are interpreted over the Cartesian products of object domains and the flow of time $(\mathbb{Z}, <)$, satisfying the constant domain assumption. Concept and role inclusions of the TBox hold at all moments of time (globally), and data assertions of the ABox hold at specified moments of time. To express temporal constraints of conceptual data models, the languages are equipped with flexible and rigid roles, standard future and past temporal operators on concepts, and operators "always" and "sometime" on roles. The most expressive of our TDLs (which can capture lifespan cardinalities and either qualitative or quantitative evolution constraints) turns out to be undecidable. However, by omitting some of the temporal operators on concepts/roles or by restricting the form of concept inclusions, we construct logics whose complexity ranges between NLogSpace and PSpace. These positive results are obtained by reduction to various clausal fragments of propositional temporal logic, which opens a way to employ propositional or first-order temporal provers for reasoning about temporal data models.

Categories and Subject Descriptors: I.2.4 [**Knowledge Representation Formalisms and Methods**]: Representation Languages; F.4.1 [**Mathematical Logic**]: Temporal Logic; F.2.2 [**Nonnumerical Algorithms and Problems**]: Complexity of Proof Procedures; H.2.1 [**Logical Design**]: Data Models

General Terms: Languages, Theory

Additional Key Words and Phrases: Description logic, temporal conceptual data model

ACM Reference Format:
Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. 2014. A cookbook for temporal conceptual data modelling with description logics. ACM Trans. Comput. Logic 15, 3, Article 25 (June 2014), 50 pages.
DOI: http://dx.doi.org/10.1145/2629565

This work was partially supported by the UK EPSRC grant EP/H05099X/1.
Authors' addresses: Dr A. Artale, KRDB Research Centre, Free University of Bozen-Bolzano, Piazza Domenicani 3, I-39100 Bozen-Bolzano BZ, Italy; email: artale@inf.unibz.it; Dr R. Kontchakov, Department of Computer Science and Information Systems, Birkbeck, University of London, Malet Street, London WC1E 7HX, UK; email: roman@dcs.bbk.ac.uk; Dr V. Ryzhikov, KRDB Research Centre, Free University of Bozen-Bolzano, Piazza Domenicani 3, I-39100 Bozen-Bolzano BZ, Italy; email: ryzhikov@inf.unibz.it; Prof M. Zakharyachev, Department of Computer Science and Information Systems, Birkbeck, University of London, Malet Street, London WC1E 7HX, UK; email: michael@dcs.bbk.ac.uk.

1. INTRODUCTION

The aim of this article is twofold. On the one hand, we investigate the complexity of reasoning about temporal conceptual data models (TCMs) depending on the available modelling constructs. On the other hand, we achieve this by encoding TCMs in carefully crafted temporal description logics (TDLs). As a result, we obtain a new family of TDLs and a clear understanding of how their constructs affect the complexity of reasoning. Most of the constructed TDLs feature an unexpectedly low complexity—compared to other known TDLs—such as NLogSpace, PTime, NP, and PSpace, which is good news for automated temporal conceptual modelling. However, some combinations of the constructs (which involve temporal operators on relationships) result in undecidability, giving a new type of undecidable fragments of first-order temporal logic.

Conceptual data modelling formalisms, such as the extended entityrelationship (EER) model and Unified Modelling Language (UML), provide visual means to describe application domains in a declarative and reusable way, and they are regarded as standard tools in database design and software engineering. One of the main tasks in conceptual modelling is to ensure that conceptual schemas satisfy various “quality properties”: for instance, one may wish to check whether a given schema is consistent, whether its entities and relationships can be populated, whether a certain individual is an instance of a certain class, and so forth. That was where conceptual modelling met description logics (DLs), a family of knowledge representation formalisms specifically designed to efficiently reason about structured knowledge [Baader et al. 2003]. Since 2007, DLs have been recognised as the backbone of the Semantic Web, underlying the standard Web Ontology Languages OWL and OWL 2.[1]

Connections between conceptual data models (CMs) and DLs have been investigated since the 1990s (e.g., see Calvanese et al. [1999], Borgida and Brachman [2003], Berardi et al. [2005], Artale et al. [2007a] and references therein), resulting in a classification of CMs according to the computational complexity of checking schema consistency depending on the available modelling constructs. The standard EER/UML constructs include generalisation (inheritance) for entities (classes), relationships and attributes with disjointness and covering constraints on them, cardinality constraints for relationships and their refinements, multiplicity constraints for attributes, and key constraints for entities. Reasoning over CMs equipped with the full set of constructs is ExpTime-complete, which was shown by mapping CMs into the DLs $\mathcal{DLR}$ and $\mathcal{ALCQI}$ [Calvanese et al. 1999; Berardi et al. 2005]. With the invention of the *DL-Lite* family [Calvanese et al. 2005, 2007; Artale et al. 2007b, 2009a], it became clear that reasoning over CMs can often be done using DLs much weaker than $\mathcal{DLR}$ and $\mathcal{ALCQI}$. For example, the NP-complete $\textit{DL-Lite}_{bool}^{(\mathcal{HN})}$ was shown to be adequate for representing a large class of CMs with generalisation and both disjointness and covering constraints, but no upper cardinality bounds on specialised relationships (see Artale et al. [2007a] and Section 2.2 for details). If we are also prepared to sacrifice covering constraints, then the NLogSpace-complete fragment $\textit{DL-Lite}_{core}^{(\mathcal{HN})}$ can do the job. (Note that $\textit{DL-Lite}_{core}^{(\mathcal{HN})}$ contains the OWL 2 QL profile[2] of OWL 2 and the DL fragment of RDF Schema, RDFS.[3])

TCMs extend CMs with means to represent constraints over temporal database instances. Temporal constraints can be grouped into three categories: *timestamping*, *evolution,* and *temporal cardinality* constraints. Timestamping constraints discriminate between those classes, relationships, and attributes that change over time and those that are time invariant (or, *rigid*) [Theodoulidis et al. 1991; Gregersen and Jensen 1999;

[1]www.w3.org/2007/OWL, www.w3.org/TR/owl2-overview.
[2]www.w3.org/TR/owl2-profiles.
[3]www.w3.org/TR/rdf-schema.

Finger and McBrien 2000; Artale and Franconi 1999; Parent et al. 2006]. Evolution constraints control how the domain elements evolve over time by migrating from one class to another [Hall and Gupta 1991; Mendelzon et al. 1994; Su 1997; Parent et al. 2006; Artale et al. 2007e]. We distinguish between qualitative evolution constraints describing generic temporal behaviour, and quantitative ones specifying the exact time of migration. Temporal cardinality constraints restrict the number of times an instance of a class can participate in a relationship: snapshot cardinality constraints do it at each moment of time, whereas lifespan cardinality constraints impose restrictions over the entire existence of the instance as a member of the class [Tauzovich 1991; McBrien et al. 1992; Artale and Franconi 2009].

Temporal extensions of DLs have been constructed and investigated since the seminal papers of Schmiedel [1990] and Schild [1993] (e.g., see Gabbay et al. [2003], Artale and Franconi [2001, 2005], and Lutz et al. [2008] for detailed surveys), with reasoning over TCMs being one of the main objectives. The first attempts to represent TCMs by means of TDLs resulted in fragments of $\mathcal{DLR}_{\mathcal{US}}$ and $\mathcal{ALCQI}_{\mathcal{US}}$ whose complexity ranged from ExpTime and ExpSpace up to undecidability [Artale and Franconi 1999; Artale et al. 2002, 2003]. A general conclusion one could draw from the obtained results is that—as far as there is a nontrivial interaction between the temporal and DL components—TDLs based on full-fledged DLs such as $\mathcal{ALC}$ turn out to be too complex for effective practical reasoning (this will be discussed in more detail in Section 3.3).

The possibility to capture CMs using logics of the *DL-Lite* family gave a glimpse of hope that automated reasoning over TCMs can finally be made practical. The first temporal extension of $\textit{DL-Lite}_{\mathit{bool}}^{(\mathcal{HN})}$ was constructed by Artale et al. [2007c]. It featured rigid roles, with temporal and Boolean operators applicable not only to concepts but also to TBox axioms and ABox assertions. The resulting logic was shown to be ExpSpace-complete. (To compare, the same temporalisation of $\mathcal{ALC}$ is trivially undecidable [Artale et al. 2002; Gabbay et al. 2003].) This encouraging result prompted a systematic investigation of TDLs suitable for reasoning about TCMs.

Our aim in this article is to design *DL-Lite*–based TDLs that are capable of representing various sets of TCM constructs and have as low computational complexity as possible. Let us first formulate our minimal requirements for such TDLs. At the model-theoretic level, we are interested in temporal interpretations that are Cartesian products of object domains and the flow of time $(\mathbb{Z}, <)$. At each moment of time, we interpret the DL constructs over the same domain (thus complying with the constant domain assumption adopted in temporal databases [Chomicki et al. 2001]). We want to be able to specify, using temporal ABoxes, that a finite number of concept and role membership assertions hold at specific moments of time. We regard timestamping constraints as indispensable; this means, in particular, that we should be able to declare that certain roles and concepts are rigid (time invariant) in the sense that their interpretations do not change over time. Other temporal and static (atemporal) modelling constraints are expressed by means of TBox axioms (concept and role inclusions). In fact, we observe that to represent TCM constraints, we only require concept and role inclusions that hold globally, at every time instant; thus, temporal and Boolean operators on TBox axioms [Artale et al. 2007c; Baader et al. 2008, 2012] are not needed for our aims (but may be useful to impose constraints on schema evolution). Finally, to represent cardinality constraints (both snapshot and lifespan), we require number restrictions; thus, we assume this construct to be available in all of our formalisms.

The remaining options include the choice of (1) the underlying dialect of *DL-Lite* for disjointness and covering constraints; (2) the temporal operators on concepts for different types of evolution constraints, and (3) the temporal operators on roles for lifespan cardinality constraints. For (1), we consider three DLs: $\textit{DL-Lite}_{\mathit{bool}}^{(\mathcal{HN})}$ and its sub-Boolean fragments $\textit{DL-Lite}_{\mathit{krom}}^{(\mathcal{HN})}$ and $\textit{DL-Lite}_{\mathit{core}}^{(\mathcal{HN})}$. For (2), we take various subsets of the standard future and past temporal operators (since and until, next and previous time, sometime and always in the future/past, or simply sometime and always). Finally, for (3), we only use the undirected temporal operators "always" and "sometime" (referring to all time instants); roles in the scope of such operators are called *temporalised*.

Our most expressive TDL, based on $\textit{DL-Lite}_{bool}^{(\mathcal{HN})}$, captures all standard types of temporal constraints: timestamping, evolution, and temporal cardinality. Unfortunately, and to our surprise, this TDL turns out to be undecidable. As follows from the proof of Theorem 6.1, it is a subtle interaction of functionality constraints on temporalised roles with the temporal operators and full Booleans on concepts that causes undecidability. On a more positive note, we show that even small restrictions of this interaction result in TDLs with better computational properties.

First, keeping $\textit{DL-Lite}_{bool}^{(\mathcal{HN})}$ as the base DL but limiting the temporal operators on concepts to "always" and "sometime," we obtain an NP-complete logic, which can express timestamping and lifespan cardinalities. To appreciate this result, recall that a similar logic based on $\mathcal{ALC}$ is 2ExpTime-complete [Artale et al. 2007d]. Second, by giving up temporalised roles but retaining temporal operators on concepts, we obtain PSpace- or NP-complete logics depending on the available temporal operators, which matches the complexity of the underlying propositional temporal logic. These TDLs have sufficient expressivity to capture timestamping and evolution constraints but cannot represent temporal cardinality constraints (see Section 3). We prove these upper complexity bounds by a reduction to the propositional temporal logic $\mathcal{PTL}$, which opens a way to employ the existing temporal provers for checking quality properties of TCMs. Again, we note that a similar logic based on $\mathcal{ALC}$ is undecidable [Wolter and Zakharyaschev 1999; Artale et al. 2002; Gabbay et al. 2003].

We can reduce the complexity even further by restricting $\textit{DL-Lite}_{bool}^{(\mathcal{HN})}$ to its sub-Boolean fragments $\textit{DL-Lite}_{krom}^{(\mathcal{HN})}$ and $\textit{DL-Lite}_{core}^{(\mathcal{HN})}$, which are unable to capture covering constraints. This results in logics within NP and PTime. And if the temporal operators on concepts are limited to "always" and "sometime," then the two sub-Boolean fragments are NLogSpace-complete. To obtain these results, we consider sub-Boolean fragments of $\mathcal{PTL}$ by imposing restrictions on both the type of clauses in Separated Normal Form (SNF) [Fisher 1991] and the available temporal operators. We give a complete classification of such fragments according to their complexity (see Table III).

The rest of the article is organised as follows. Section 2 introduces, using a simple example, conceptual data modelling languages and illustrates how they can be captured by various dialects of *DL-Lite*, which are formally defined in Section 2.2. Section 3 introduces temporal conceptual modelling constraints using a temporal extension of our example. In Section 3.2, we design *DL-Lite*–based TDLs that can represent those constraints. Section 3.3 gives a detailed overview of the results obtained in this article together with a discussion of related work. Section 4 gives the reduction of TDLs to $\mathcal{PTL}$ mentioned earlier. In Section 5, we establish the complexity results for the clausal fragments of propositional temporal logic. Section 6 studies the complexity of TDLs with temporalised roles. We discuss the obtained results, open problems, and future directions in Section 7.

2. CONCEPTUAL MODELLING AND DESCRIPTION LOGIC

DLs (e.g., see Baader et al. [2003]) were designed in the 1980s as logic-based formalisms for knowledge representation and reasoning; their major application areas include ontologies in life sciences and the Semantic Web. Conceptual modelling languages [Chen 1976] are a decade older and were developed for abstract data representation in database design. Despite apparent notational differences, both families of languages are built around concepts (or entities) and relationships using a number of "natural" constructs; a close correspondence between them was discovered and investigated in

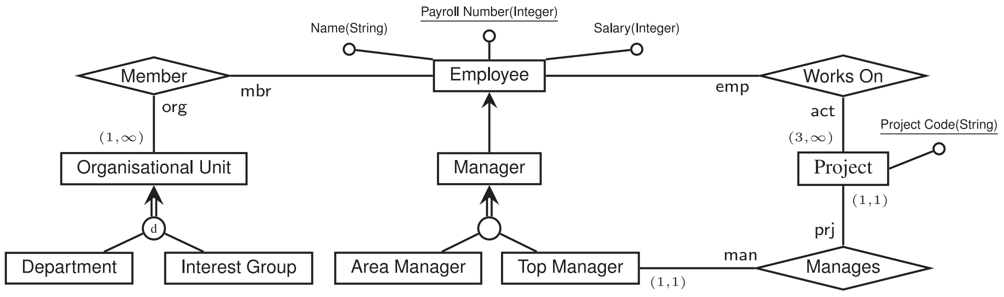

Fig. 1. A conceptual data model of a company information system.

Calvanese et al. [1999], Borgida and Brachman [2003], Berardi et al. [2005], and Artale et al. [2007a].

The *DL-Lite* DLs [Calvanese et al. 2005, 2007; Poggi et al. 2008; Artale et al. 2007b, 2009a] and the *DL-Lite*-based profile OWL 2 QL of OWL 2 have grown from the idea of linking relational databases and ontologies in the framework of ontology-based data access (OBDA) [Dolby et al. 2008; Heymans et al. 2008; Poggi et al. 2008]. The chief aims that determined the shape of the *DL-Lite* logics are (i) the ability to represent basic constraints used in conceptual modelling and (ii) the ability to support query answering using standard relational database systems. In this article, we concentrate on *DL-Lite* as a modelling language and briefly return to the issue of OBDA in Section 7.

In this section, we give an intuitive example illustrating the main constructs of conceptual data models and their *DL-Lite* representations. In the example, we use the EER language [Elmasri and Navathe 2007]; however, one can easily employ other conceptual modelling formalisms such as UML class diagrams (www.uml.org). Then we formally define the syntax and semantics of the *DL-Lite* logics to be used later on in this article.

2.1. A Motivating Example

Let us consider the EER diagram in Figure 1 representing (part of) a company information system. The arrow from the entity Manager to the entity Employee stands for the statement "all managers are employees." The double arrow with a circle below Manager means that the set of managers is the union of the set of area managers and the set of top managers. These statements can be represented in the language of DL as inclusions between concepts:

$$\begin{aligned} &\mathit{Manager} \sqsubseteq \mathit{Employee}, && \mathit{AreaManager} \sqsubseteq \mathit{Manager},\\ &\mathit{Manager} \sqsubseteq \mathit{AreaManager} \sqcup \mathit{TopManager}, && \mathit{TopManager} \sqsubseteq \mathit{Manager}.\end{aligned}$$

Here, *Manager*, *Employee*, *AreaManager*, and *TopManager* are *concept names* (or unary predicates), and the symbols $\sqsubseteq$ and $\sqcup$ denote the usual set-theoretic inclusion and union, respectively. In a similar way, we read and represent the part of the EER diagram located below Organisational Unit; the only new ingredient here is the circled d, indicating that the union is *disjoint*:

$$\begin{aligned} &\mathit{Department} \sqsubseteq \mathit{OrganisationalUnit}, && \mathit{OrganisationalUnit} \sqsubseteq \mathit{Department} \sqcup \mathit{InterestGroup},\\ &\mathit{InterestGroup} \sqsubseteq \mathit{OrganisationalUnit}, && \mathit{Department} \sqcap \mathit{InterestGroup} \sqsubseteq \bot.\end{aligned}$$

Here, $\bot$ denotes the empty set and $\sqcap$ the set-theoretic intersection.

The entity Employee in Figure 1 has three attributes: Name, which is a string, and Payroll Number and Salary, both of which are integers. The attribute Payroll Number (underlined) is a *key* for the entity Employee. In DL, we can encode attributes by means of *roles* (binary predicates). For example, to say that every employee has a salary, which is an integer number, we can represent the attribute Salary by a role, *salary*, together with the concept inclusions

$$Employee \sqsubseteq \exists salary, \qquad\qquad \exists salary^- \sqsubseteq Integer,$$

where $\exists salary$ denotes the domain of *salary*, and $salary^-$ is the inverse of *salary* so that $\exists salary^-$ is the range of *salary*. Then, the fact that each individual has a unique *salary* attribute value can be expressed by the concept inclusion

$$\geq 2\, salary \sqsubseteq \bot,$$

where $\geq 2\, salary$ stands for the set of all domain elements with at least two values of *salary* attached to them (which must be empty according to this inclusion, i.e., *salary* is a *functional* role). The attributes Payroll Number and Name are represented in a similar manner. The fact that Payroll Number is a key for Employee can be encoded by the inclusion

$$\geq 2\, payrollNumber^- \sqsubseteq \bot.$$

Relationships are used to describe connections among objects from (possibly) different entities. Works On, Member, and Manages in Figure 1 are binary relationships. The argument emp of Works On is of type Employee in the sense that its values always belong to the entity Employee (in other words, Employee participates in Works On as emp). Likewise, the argument act of Works On is of type Project. In DL, a binary relationship such as Works On can be represented by a role, say, *worksOn*. If we agree that the first argument of *worksOn* corresponds to emp and the second to act, then the domain of *worksOn* belongs to *Employee* and its range to *Project*:

$$\exists worksOn \sqsubseteq Employee, \qquad\qquad \exists worksOn^- \sqsubseteq Project.$$

The expression $(3, \infty)$ labelling the argument act of Works On is a *cardinality constraint*, meaning that every element of the set Project participates in at least three distinct tuples in the relationship Works On (each project involves at least three employees). This can be represented by the inclusion

$$Project \sqsubseteq \geq 3\, worksOn^-. \tag{1}$$

The expression $(1, 1)$ labelling the argument prj of the relationship Manages means that each element of Project participates in at least one and at most one—that is, exactly one—tuple in Manages, which is represented by two inclusions:

$$Project \sqsubseteq \exists manages^-, \qquad\qquad Project \sqsubseteq \leq 1\, manages^-.$$

Relationships of arity greater than 2 are encoded by using *reification* [Calvanese et al. 2001] (binary relationships can also be reified). For instance, to reify the binary relationship Works On, we introduce a new concept name, say *C-WorksOn*, and two functional roles, *emp* and *act*, satisfying the following concept inclusions:

$$C\text{-}WorksOn \sqsubseteq \exists emp, \quad \geq 2\, emp \sqsubseteq \bot, \quad \exists emp \sqsubseteq C\text{-}WorksOn, \quad \exists emp^- \sqsubseteq Employee, \tag{2}$$

$$C\text{-}WorksOn \sqsubseteq \exists act, \quad \geq 2\, act \sqsubseteq \bot, \quad \exists act \sqsubseteq C\text{-}WorksOn, \quad \exists act^- \sqsubseteq Project. \tag{3}$$

Thus, each element of *C-WorksOn* is related, via the roles *emp* and *act*, to a unique pair of elements of *Employee* and *Project*. Cardinality constraints are still representable for reified relations—for example, the cardinality expressed by the formula (1) becomes

$$Project \sqsubseteq \geq 3\, act^-. \tag{4}$$

Of the data modelling constructs not used in Figure 1, we mention here *relationship generalisation*—that is, a possibility to state that one relationship is a subset of another relationship. For example, we can state that everyone managing a project must also work on the project. In other words: Manages is a subrelationship of Works On, which can be represented in DL as the role inclusion

$$\mathit{manages} \sqsubseteq \mathit{worksOn}$$

if both relationships are binary and not reified. On the other hand, if both relationships are reified, then we need a concept inclusion between the respective reifying concepts as well as role inclusions between the functional roles for their arguments:

$$\mathit{C\text{-}Manages} \sqsubseteq \mathit{C\text{-}WorksOn}, \qquad \mathit{prj} \sqsubseteq \mathit{act}, \qquad \mathit{man} \sqsubseteq \mathit{emp}.$$

To represent database instances of a conceptual model, we use assertions such as $\mathit{Manager}(\mathit{bob})$ for "Bob is a manager" and $\mathit{manages}(\mathit{bob}, \mathit{cronos})$ for "Bob manages Cronos."

As conceptual data models can be large and contain nontrivial implicit knowledge, it is important to make sure that the constructed conceptual model satisfies certain *quality properties*. For example, one may want to know whether it is consistent, whether all or some of its entities and relationships are not necessarily empty or whether one entity or relationship is (not) subsumed by another. To automatically check such quality properties, it is essential to provide an effective reasoning support during the construction phase of a conceptual model.

We now define the reasoning problems formally, by giving the syntax and semantics of DLs containing the constructs discussed previously.

2.2. *DL-Lite* Logics

We start with the logic called $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ in the nomenclature of Artale et al. [2009a]. The language of $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ contains *object names* $a_0, a_1, \dots,$ *concept names* $A_0, A_1, \dots,$ and *role names* $P_0, P_1, \dots$. *Roles* R, *basic concepts* B, and *concepts* C of this language are defined by the grammar:

$$\begin{array}{lllllll} R & ::= & P_k & \mid & P_k^-, & & \\ B & ::= & \bot & \mid & A_k & \mid & \geq q\, R, \\ C & ::= & B & \mid & \neg C & \mid & C_1 \sqcap C_2, \end{array}$$

where q is a positive integer represented in binary. A $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ TBox, $\mathcal{T}$, is a finite set of concept inclusion axioms of the form

$$C_1 \sqsubseteq C_2.$$

An ABox, $\mathcal{A}$, is a finite set of assertions of the form

$$A_k(a_i), \qquad \neg A_k(a_i), \qquad P_k(a_i, a_j), \qquad \neg P_k(a_i, a_j).$$

Taken together, $\mathcal{T}$ and $\mathcal{A}$ constitute the knowledge base (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$.

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of this and other *DL-Lite* languages consists of a domain $\Delta^{\mathcal{I}} \neq \emptyset$ and an interpretation function $\cdot^{\mathcal{I}}$ that assigns to each object name a_i an element $a_i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, to each concept name A_k a subset $A_k^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and to each role name P_k a binary relation $P_k^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. As in databases, we adopt the unique name assumption (UNA): $a_i^{\mathcal{I}} \neq a_j^{\mathcal{I}}$ for all $i \neq j$ (note, however, that OWL does not use the UNA). The role and concept constructs are interpreted in $\mathcal{I}$ as follows:

$$\begin{array}{lr} (P_k^-)^{\mathcal{I}} = \big\{(y, x) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (x, y) \in P_k^{\mathcal{I}}\big\}, & \bot^{\mathcal{I}} = \emptyset, \\ (\geq q\, R)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \in \Delta^{\mathcal{I}} \mid (x, y) \in R^{\mathcal{I}}\} \geq q\big\}, & (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, \\ & (C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}, \end{array}$$

where $\sharp X$ denotes the cardinality of X. We use the standard abbreviations:

$$C_1 \sqcup C_2 = \neg(\neg C_1 \sqcap \neg C_2), \qquad \top = \neg\bot, \qquad \exists R = (\geq 1\, R), \qquad \leq q\, R = \neg(\geq q+1\, R).$$

Concepts of the form $\leq q\, R$ and $\geq q\, R$ are called *number restrictions*, and those of the form $\exists R$ are called *existential concepts*.

The satisfaction relation $\models$ is defined as expected:

$$\begin{array}{rlcrl} \mathcal{I} \models C_1 \sqsubseteq C_2 & \text{iff} \quad C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}, & & & \\ \mathcal{I} \models A_k(a_i) & \text{iff} \quad a_i^{\mathcal{I}} \in A_k^{\mathcal{I}}, & & \mathcal{I} \models P_k(a_i, a_j) & \text{iff} \quad (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \in P_k^{\mathcal{I}}, \\ \mathcal{I} \models \neg A_k(a_i) & \text{iff} \quad a_i^{\mathcal{I}} \notin A_k^{\mathcal{I}}, & & \mathcal{I} \models \neg P_k(a_i, a_j) & \text{iff} \quad (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \notin P_k^{\mathcal{I}}. \end{array}$$

A KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is said to be *satisfiable* (or consistent) if there is an interpretation $\mathcal{I}$ satisfying all members of $\mathcal{T}$ and $\mathcal{A}$. In this case, we write $\mathcal{I} \models \mathcal{K}$ (as well as $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$) and say that $\mathcal{I}$ is a *model of* $\mathcal{K}$ (and of $\mathcal{T}$ and $\mathcal{A}$). The satisfiability problem—given a KB $\mathcal{K}$, decide whether $\mathcal{K}$ is satisfiable—is the main reasoning problem that we consider in this article. Subsumption (given an inclusion $C_1 \sqsubseteq C_2$ and a TBox $\mathcal{T}$, decide whether $\mathcal{I} \models C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$; or $\mathcal{T} \models C_1 \sqsubseteq C_2$ in symbols) and concept satisfiability (given a concept C and a TBox $\mathcal{T}$, decide whether there is a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$; or $\mathcal{T} \not\models C \sqsubseteq \bot$) are reducible to satisfiability. For example, to check whether $\mathcal{T} \models C_1 \sqsubseteq C_2$, we can construct a new KB $\mathcal{K} = (\mathcal{T} \cup \{A \sqsubseteq C_1, A \sqsubseteq \neg C_2\}, \{A(a)\})$ with a fresh concept name A and check whether $\mathcal{K}$ is *not* satisfiable.

The two sublanguages of $\mathit{DL\text{-}Lite}_{bool}^{\mathcal{N}}$ that we deal with in this article are obtained by restricting the Boolean operators on concepts. In $\mathit{DL\text{-}Lite}_{krom}^{\mathcal{N}}$ TBoxes,[4] concept inclusions are of the form

$$B_1 \sqsubseteq B_2, \qquad B_1 \sqsubseteq \neg B_2 \qquad \text{or} \qquad \neg B_1 \sqsubseteq B_2. \tag{krom}$$

(Here and later, B_1, B_2 are basic concepts.) $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$ only uses concept inclusions of the form

$$B_1 \sqsubseteq B_2 \qquad \text{or} \qquad B_1 \sqcap B_2 \sqsubseteq \bot. \tag{core}$$

As $B_1 \sqsubseteq \neg B_2$ is equivalent to $B_1 \sqcap B_2 \sqsubseteq \bot$, $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$ is a sublanguage of $\mathit{DL\text{-}Lite}_{krom}^{\mathcal{N}}$. Although the Krom fragment does not seem to be more useful for conceptual modelling than $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$, we shall see in Remark 3.2 that temporal extensions of $\mathit{DL\text{-}Lite}_{krom}^{\mathcal{N}}$ can capture some important temporal modelling constructs that are not representable by the corresponding extensions of $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$.

Most of the constraints in the company conceptual model from Section 2.1 were represented by means of $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$ concept inclusions. The only exceptions were the covering constraints $\mathit{Manager} \sqsubseteq \mathit{AreaManager} \sqcup \mathit{TopManager}$ and $\mathit{OrganisationalUnit} \sqsubseteq \mathit{Department} \sqcup \mathit{InterestGroup}$, which belong to the language $\mathit{DL\text{-}Lite}_{bool}^{\mathcal{N}}$, and the role inclusion $\mathit{manages} \sqsubseteq \mathit{worksOn}$. The extra expressive power, gained from the addition of covering constraints to $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$, comes at a price [Artale et al. 2007b]: the satisfiability problem is NLogSpace-complete for $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$ and $\mathit{DL\text{-}Lite}_{krom}^{\mathcal{N}}$ KBs and NP-complete for $\mathit{DL\text{-}Lite}_{bool}^{\mathcal{N}}$ KBs.

The straightforward extension of $\mathit{DL\text{-}Lite}_{core}^{\mathcal{N}}$ with role inclusions of the form

$$R_1 \sqsubseteq R_2 \qquad (\text{with} \quad \mathcal{I} \models R_1 \sqsubseteq R_2 \quad \text{iff} \quad R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}})$$

leads to an even higher complexity: satisfiability becomes ExpTime-complete [Artale et al. 2009a]. The reason for this is the interaction of functionality constraints and role

[4]The Krom fragment of first-order logic consists of formulas in prenex normal form whose quantifier-free part is a conjunction of binary clauses.

Table I. Complexity of the *DL-Lite* Logics

	role inclusions		
concept inclusions	$DL\text{-}Lite_\alpha^{\mathcal{N}}$	$DL\text{-}Lite_\alpha^{(\mathcal{HN})}$	$DL\text{-}Lite_\alpha^{\mathcal{HN}}$
Bool	NP	NP	ExpTime
Krom	NLogSpace	NLogSpace	ExpTime
core	NLogSpace	NLogSpace	ExpTime

inclusions such as

$$R_1 \sqsubseteq R_2 \quad \text{and} \quad {\geq}2\, R_2 \sqsubseteq \bot.$$

Note that inclusions of this sort are required when we use relationship generalisation with reification (see Section 2.1). If we restrict this interaction in TBoxes $\mathcal{T}$ by requiring that no role R can occur in $\mathcal{T}$ in both a role inclusion of the form $R' \sqsubseteq R$ and a number restriction ${\geq}q\, R$ or ${\geq}q\, R^-$ with $q \geq 2$, then the complexity of satisfiability checking with such TBoxes matches that of the language without role inclusions. The extension of $DL\text{-}Lite_\alpha^{\mathcal{N}}$, where $\alpha \in \{core, krom, bool\}$, with role inclusions satisfying the condition shown earlier is denoted by $DL\text{-}Lite_\alpha^{(\mathcal{HN})}$; without this condition, the extension is denoted by $DL\text{-}Lite_\alpha^{\mathcal{HN}}$. Table I summarises the complexity of the KB satisfiability problem for *DL-Lite* logics (for details, consult Artale et al. [2009a]).

Thus, already in the atemporal case, a conceptual data model engineer has to search for a suitable compromise between the expressive power of the modelling language and efficiency of reasoning. In the temporal case, the trade-off between expressiveness and efficiency becomes even more dramatic.

In the next section, we extend the atemporal conceptual data model considered earlier with a number of temporal constructs and use them to design a family of TDLs that are suitable for temporal conceptual modelling.

3. TEMPORAL CONCEPTUAL MODELLING AND TEMPORAL DESCRIPTION LOGIC

TCMs extend standard conceptual schemas with means to visually represent temporal constraints imposed on temporal database instances [Theodoulidis et al. 1991; Tauzovich 1991; Jensen and Snodgrass 1999; Artale et al. 2003; Parent et al. 2006; Combi et al. 2008].

When introducing a temporal dimension into conceptual data models, time is usually modelled by a linearly ordered set of time instants; thus, at each moment of time, we can refer to its past and future. In this article, we assume that the flow of time is isomorphic to the strictly linearly ordered set $(\mathbb{Z}, <)$ of integer numbers. (For a survey of other options, including interval-based and branching models of time, consult, e.g., Gabbay et al. [1994, 2000, 2003].)

We will now introduce the most important temporal conceptual modelling constructs by extending the company information system example from Section 2.1.

3.1. The Motivating Example Temporalised

A basic assumption in temporal conceptual models is that entities, relationships, and attributes may freely change over time as long as they satisfy the constraints of the schema at *each* time instant. Temporal constructs are used to impose constraints on the temporal behaviour of various components of conceptual schemas. We group these constructs into three categories—*timestamping*, *evolution*, and *temporal cardinality constraints*—and illustrate them by the model in Figure 2.

Timestamping constraints [Theodoulidis et al. 1991; Gregersen and Jensen 1998, 1999; Finger and McBrien 2000; Artale and Franconi 1999; Parent et al. 2006] distinguish between entities, relationships, and attributes that are

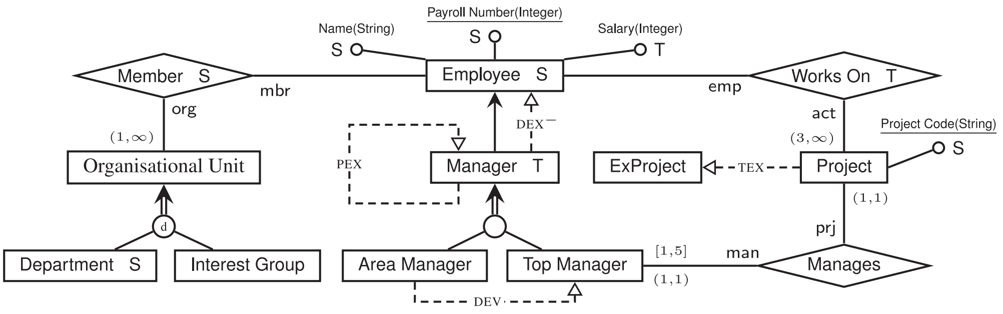

Fig. 2. A temporal conceptual model of a company information system.

—*temporary* in the sense that no element belongs to them at all moments of time,
—*snapshot*, or time invariant, in the sense that their interpretation does not change with time,
—*unconstrained* (all others).

In temporal entity-relationship diagrams, the temporary entities, relationships, and attributes are marked with T and the snapshot ones with S. In Figure 2, Employee and Department are snapshot entities; Name, Payroll Number, and Project Code are snapshot attributes; and Member a snapshot relationship. On the other hand, Manager is a temporary entity, Salary a temporary attribute, and Works On a temporary relationship.

There are (at least) two ways of representing timestamping constraints in TDLs. One of them is to introduce special names for temporary and snapshot concepts and roles, and interpret them accordingly. Another way is to employ a temporal operator $\boxast$, which is read as "always" or "at all—past, present, and future—time instants." Intuitively, for a concept C, $\boxast C$ contains those elements that belong to C at all time instants. Using this operator, the constraints "Employee is a snapshot entity" and "Manager is a temporary entity" can be represented as follows:

$$\mathit{Employee} \sqsubseteq \boxast \mathit{Employee}, \qquad\qquad \boxast \mathit{Manager} \sqsubseteq \bot.$$

The first inclusion says that at any moment of time, every element of Employee has always been and will always be an element of Employee. The second one states that no element can belong to Manager at all time instants. Note that both of these concept inclusions are meant to hold *globally*—that is, at all moments of time.

The same temporal operator $\boxast$ together with rigid roles (i.e., roles that do not change over time) can be used to capture timestamping of reified relationships. If the relationship Member is reified by the concept *C-Member* with two functional roles *org* and *mbr*, satisfying the concept inclusions similar to (2) and (3), then the requirement that both roles *org* and *mbr* are rigid ensure that Member is a snapshot relationship. On the other hand, for the reified temporary relationship Works On, we require the concept inclusion

$$\boxast \mathit{C\text{-}WorksOn} \sqsubseteq \bot$$

and two flexible roles *emp* and *act*, which can change arbitrarily. Rigid roles are also used to represent both snapshot attributes and snapshot binary relationships. Temporary attributes can be captured by flexible roles or by using temporalised roles:

$$\exists \boxast \mathit{salary} \sqsubseteq \bot,$$

where $\boxast \mathit{salary}$ denotes the intersection of the relations *salary* at all time instants.

Evolution constraints control how the domain elements evolve over time by "migrating" from one entity to another [Hall and Gupta 1991; Mendelzon et al. 1994; Su

1997; Artale et al. 2007e]. We distinguish between *qualitative* evolution constraints that describe generic temporal behaviour but do not specify the moment of migration, and *quantitative* evolution (or transition) constraints that specify the exact moment of migration. The dashed arrow marked with TEX (TRansition EXtension[5]) in Figure 2 is an example of a quantitative evolution constraint, meaning that each project expires in exactly one time unit (one year) and becomes an instance of ExProject. The dashed arrow marked with DEV (DYnamic EVolution) is a qualitative evolution constraint meaning that every area manager will eventually become a top manager. The DEX$^-$ (DYnamic EXtension) dashed arrow says that every manager was once an employee, whereas the PEX (PErsistent EXtension) dashed arrow means that a manager will always be a manager and cannot be demoted.

In TDL, these evolution constraints are represented using temporal operators such as "at the next moment of time" $\bigcirc_F$, "sometime in the future" $\Diamond_F$, "sometime in the past" $\Diamond_P$, and "always in the future" $\Box_F$:

$$\begin{aligned} \mathit{Project} &\sqsubseteq \bigcirc_F \mathit{ExProject}, & \mathit{AreaManager} &\sqsubseteq \Diamond_F \mathit{TopManager},\\ \mathit{Manager} &\sqsubseteq \Diamond_P \mathit{Employee}, & \mathit{Manager} &\sqsubseteq \Box_F \mathit{Manager}. \end{aligned}$$

Again, these concept inclusions must hold globally. In the following, the evolution constraints that involve $\Diamond_F$ and $\Diamond_P$ will be called *migration constraints*.

Temporal cardinality constraints [Tauzovich 1991; McBrien et al. 1992; Gregersen and Jensen 1998] restrict the number of times an instance of an entity participates in a relationship. Snapshot cardinality constraints do that at each moment of time, whereas lifespan cardinality constraints impose restrictions over the entire existence of the instance as a member of the entity. In Figure 2, we use (k, l) to specify the snapshot cardinalities and $[k, l]$ the lifespan cardinalities: for example, at any moment, every top manager manages exactly one project, but not more than five different projects over the whole career. If the relationship Manages is not reified and represented by a role in TDL, then these two constraints can be expressed by the following concept inclusions:

$$\mathit{TopManager} \sqsubseteq \exists \mathit{manages} \sqcap \leq 1\, \mathit{manages}, \qquad \mathit{TopManager} \sqsubseteq\ \leq 5 \diamondplus \mathit{manages},$$

where $\diamondplus$ means "sometime" (in the past, present, or future), and so $\diamondplus \mathit{manages}$ is the union of the relations *manages* over *all* time instants. Snapshot and lifespan cardinalities can also be expressed in a similar way even for reified relationships (e.g., see (4), which captures snapshot cardinalities). Observe that the preceding inclusions imply, in particular, that no one can remain a top manager for longer than 5 years (indeed, each top manager manages at least one project a year, each project expires in a year, and no top manager can manage more than five projects throughout the lifetime). However, this is inconsistent with "every manager always remains a manager," and so the entity Manager cannot be populated by instances, which, in turn, means that Project must also be empty (since each project is managed by a top manager). One can make these entities consistent by, for example, dropping the persistence constraint on Manager or the upper lifespan cardinality bound on the number of projects a top manager can manage throughout the lifetime. In large schemas, situations like this can easily remain undetected if the quality check is performed manually.

To represent temporal database instances, we use assertions like $\bigcirc_P \mathit{Manager}(\mathit{bob})$ for "Bob was a manager last year" and $\bigcirc_F \mathit{manages}(\mathit{bob}, \mathit{cronos})$ for "Bob will manage Cronos next year."

[5]We refer to Artale et al. [2010] for a detailed explanation of the various evolution constraints and their naming convention.

3.2. Temporal *DL-Lite* Logics

It is known from temporal logic [Gabbay et al. 1994] that all of the temporal operators used in the previous section can be expressed in terms of the binary operators $\mathcal{S}$ "since" and $\mathcal{U}$ "until" (details will be given later). So we formulate our "base" temporal extension $T_{\mathcal{US}}\textit{DL-Lite}^{\mathcal{N}}_{bool}$ of the DL $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ using only these two operators. The language of $T_{\mathcal{US}}\textit{DL-Lite}^{\mathcal{N}}_{bool}$ contains *object names* $a_0, a_1, \dots$, *concept names* $A_0, A_1, \dots$, *flexible role names* $P_0, P_1, \dots$, and *rigid role names* $G_0, G_1, \dots$. *Role names* S and *roles* R are defined by taking

$$S \ ::= \ P_i \ \mid \ G_i \qquad \text{and} \qquad R \ ::= \ S \ \mid \ S^-.$$

We say that R is a rigid role if it is of the form G_i or G_i^-, for a rigid role name G_i. Basic concepts B, concepts C, and temporal concepts D are given by the following grammar:

$$\begin{aligned} B \ &::= \ \bot \ \mid \ A_i \ \mid \ \geq q\, R, \\ C \ &::= \ B \ \mid \ D \ \mid \ \neg C \ \mid \ C_1 \sqcap C_2, \\ D \ &::= \ C \ \mid \ C_1 \,\mathcal{U}\, C_2 \mid \ C_1 \,\mathcal{S}\, C_2, \end{aligned}$$

where, as before, q is a positive integer given in binary. (We use two separate rules for C and D here because in the definitions of the fragments of $T_{\mathcal{US}}\textit{DL-Lite}^{\mathcal{N}}_{bool}$ shown later, these rules will be restricted to the corresponding sub-Boolean and temporal fragments.) A $T_{\mathcal{US}}\textit{DL-Lite}^{\mathcal{N}}_{bool}$ *TBox*, $\mathcal{T}$, is a finite set of concept inclusions of the form $C_1 \sqsubseteq C_2$. An ABox, $\mathcal{A}$, consists of assertions of the form

$$\bigcirc^n A_k(a_i), \qquad \bigcirc^n \neg A_k(a_i), \qquad \bigcirc^n S(a_i, a_j), \quad \text{and} \quad \bigcirc^n \neg S(a_i, a_j),$$

where A_k is a concept name, S a (flexible or rigid) role name, a_i, a_j object names, and, for $n \in \mathbb{Z}$,

$$\bigcirc^n = \underbrace{\bigcirc_F \cdots \bigcirc_F}_{n \text{ times}}, \ \text{ if } n \geq 0, \qquad \text{and} \quad \bigcirc^n = \underbrace{\bigcirc_P \cdots \bigcirc_P}_{-n \text{ times}}, \ \text{ if } n < 0.$$

Note that we use $\bigcirc^n$ as an abbreviation and take the size of $\bigcirc^n$ to be n (in other words, the numbers n in ABox assertions are given in unary). Taken together, the TBox $\mathcal{T}$ and ABox $\mathcal{A}$ form the KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$.

A *temporal interpretation* is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}(n)})$, where $\Delta^{\mathcal{I}}$ is a nonempty interpretation domain and $\mathcal{I}(n)$ gives a standard DL interpretation for each time instant $n \in \mathbb{Z}$:

$$\mathcal{I}(n) = \big(\Delta^{\mathcal{I}}, a_0^{\mathcal{I}}, \dots, A_0^{\mathcal{I}(n)}, \dots, P_0^{\mathcal{I}(n)}, \dots, G_0^{\mathcal{I}}, \dots\big).$$

We assume, however, that the domain $\Delta^{\mathcal{I}}$ and the interpretations $a_i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ of object names and $G_i^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ of rigid role names are the same for all $n \in \mathbb{Z}$. (For a discussion of the constant domain assumption, consult Gabbay et al. [2003]. Recall also that we adopt the UNA.) The interpretations $A_i^{\mathcal{I}(n)} \subseteq \Delta^{\mathcal{I}}$ of concept names and $P_i^{\mathcal{I}(n)} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ of flexible role names can vary. The atemporal constructs are interpreted in $\mathcal{I}(n)$ as before; we write $C^{\mathcal{I}(n)}$ for the extension of C in $\mathcal{I}(n)$. The interpretation of the temporal operators is as follows:

$$(C_1 \,\mathcal{U}\, C_2)^{\mathcal{I}(n)} = \bigcup_{k>n} \Big(C_2^{\mathcal{I}(k)} \cap \bigcap_{n<m<k} C_1^{\mathcal{I}(m)} \Big),$$

$$(C_1 \,\mathcal{S}\, C_2)^{\mathcal{I}(n)} = \bigcup_{k<n} \Big(C_2^{\mathcal{I}(k)} \cap \bigcap_{n>m>k} C_1^{\mathcal{I}(m)} \Big).$$

Thus, for example, $x \in (C_1 \,\mathcal{U}\, C_2)^{\mathcal{I}(n)}$ iff there is a moment $k > n$ such that $x \in C_2^{\mathcal{I}(k)}$ and $x \in C_1^{\mathcal{I}(m)}$, for all moments m between n and k. Note that the operators $\mathcal{S}$ and $\mathcal{U}$ (as well as the $\Box$ and $\Diamond$ operators to be defined later) are "strict" in the sense that their semantics does not include the current moment of time. The nonstrict operators, which include the current moment, are obviously definable in terms of the strict ones.

As noted earlier, for the aims of TCM, it is enough to interpret concept inclusions in $\mathcal{I}$ globally:

$$\mathcal{I} \models C_1 \sqsubseteq C_2 \quad \text{iff} \quad C_1^{\mathcal{I}(n)} \subseteq C_2^{\mathcal{I}(n)} \quad \text{for } all \ n \in \mathbb{Z}.$$

A Box assertions are interpreted relatively to the initial moment, 0. Thus, we set the following:

$$\begin{array}{llllll} \mathcal{I} \models \bigcirc^n A_k(a_i) & \text{iff} & a_i^{\mathcal{I}} \in A_k^{\mathcal{I}(n)}, & \mathcal{I} \models \bigcirc^n S(a_i, a_j) & \text{iff} & (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \in S^{\mathcal{I}(n)}, \\ \mathcal{I} \models \bigcirc^n \neg A_k(a_i) & \text{iff} & a_i^{\mathcal{I}} \notin A_k^{\mathcal{I}(n)}, & \mathcal{I} \models \bigcirc^n \neg S(a_i, a_j) & \text{iff} & (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \notin S^{\mathcal{I}(n)}. \end{array}$$

We call $\mathcal{I}$ a *model* of a KB $\mathcal{K}$ and write $\mathcal{I} \models \mathcal{K}$ if $\mathcal{I}$ satisfies all elements of $\mathcal{K}$. $\mathcal{K}$ is *satisfiable* if it has a model. A concept C (role R) is satisfiable with respect to $\mathcal{K}$ if there are a model $\mathcal{I}$ of $\mathcal{K}$ and $n \in \mathbb{Z}$ such that $C^{\mathcal{I}(n)} \neq \emptyset$ (respectively, $R^{\mathcal{I}(n)} \neq \emptyset$). It is readily seen that the concept and role satisfiability problems are equivalent to KB satisfiability.

We now define a few fragments and extensions of the base language $T_{\mathcal{US}}$*DL-Lite*$_{bool}^{\mathcal{N}}$. Recall that to say C is a snapshot concept, we need the "always" operator $\boxast$ with the following meaning:

$$(\boxast\, C)^{\mathcal{I}(n)} = \bigcap_{k \in \mathbb{Z}} C^{\mathcal{I}(k)}.$$

The dual operator "sometime" is defined as usual: $\diamondast\, C = \neg \boxast\, \neg C$. In terms of $\mathcal{S}$ and $\mathcal{U}$, it can be represented as $\diamondast\, C = \top \,\mathcal{U}\, (\top \,\mathcal{S}\, C)$. Let T_U*DL-Lite*$_{bool}^{\mathcal{N}}$ be the sublanguage of $T_{\mathcal{US}}$*DL-Lite*$_{bool}^{\mathcal{N}}$ the temporal concepts D in which are of the form

$$D \ ::= \ C \ \mid \ \boxast\, C. \tag{U}$$

Thus, in T_U*DL-Lite*$_{bool}^{\mathcal{N}}$, we can express timestamping constraints (see Section 3.1).

The temporal operators $\Diamond_F$ ("sometime in the future") and $\Diamond_P$ ("sometime in the past") that are required for qualitative evolution constraints with the standard temporal logic semantics

$$(\Diamond_F C)^{\mathcal{I}(n)} = \bigcup_{k > n} C^{\mathcal{I}(k)} \quad \text{and} \quad (\Diamond_P C)^{\mathcal{I}(n)} = \bigcup_{k < n} C^{\mathcal{I}(k)}$$

can be expressed via $\mathcal{U}$ and $\mathcal{S}$ as $\Diamond_F C = \top \,\mathcal{U}\, C$ and $\Diamond_P C = \top \,\mathcal{S}\, C$; the operators $\Box_F$ ("always in the future") and $\Box_P$ ("always in the past") are defined as dual to $\Diamond_F$ and $\Diamond_P$: $\Box_F C = \neg \Diamond_F \neg C$ and $\Box_P C = \neg \Diamond_P \neg C$. We define the fragment T_{FP}*DL-Lite*$_{bool}^{\mathcal{N}}$ of $T_{\mathcal{US}}$*DL-Lite*$_{bool}^{\mathcal{N}}$ by restricting the temporal concepts D to the form

$$D \ ::= \ C \ \mid \ \Box_F C \ \mid \ \Box_P C. \tag{FP}$$

Clearly, we have the following equivalences:

$$\boxast\, C = \Box_F \Box_P C \qquad \text{and} \qquad \diamondast\, C = \Diamond_F \Diamond_P C.$$

In what follows, these equivalences will be regarded as definitions for $\boxast$ and $\diamondast$ in those languages where they are not explicitly present. Thus, T_{FP}*DL-Lite*$_{bool}^{\mathcal{N}}$ is capable of expressing both timestamping and qualitative (but not quantitative) evolution constraints.

The temporal operators $\bigcirc_F$ ("next time") and $\bigcirc_P$ ("previous time"), used in quantitative evolution constraints, can be defined as $\bigcirc_F C = \bot\,\mathcal{U}\,C$ and $\bigcirc_P C = \bot\,\mathcal{S}\,C$ so that we have

$$(\bigcirc_F C)^{\mathcal{I}(n)} = C^{\mathcal{I}(n+1)} \quad \text{and} \quad (\bigcirc_P C)^{\mathcal{I}(n)} = C^{\mathcal{I}(n-1)}.$$

The fragment of $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ with temporal concepts of the form

$$D \ ::= \ C \ \mid \ \Box_F C \ \mid \ \Box_P C \ \mid \ \bigcirc_F C \ \mid \ \bigcirc_P C \qquad \text{(FPX)}$$

will be denoted by T_{FPX}*DL-Lite*$^{\mathcal{N}}_{bool}$. In this fragment, we can express timestamping, and qualitative and quantitative evolution constraints.

Thus, we have the following inclusions between the languages introduced earlier:

$$T_U\textit{DL-Lite}^{\mathcal{N}}_{bool} \ \subseteq \ T_{FP}\textit{DL-Lite}^{\mathcal{N}}_{bool} \ \subseteq \ T_{FPX}\textit{DL-Lite}^{\mathcal{N}}_{bool} \ \subseteq \ T_{\mathcal{US}}\textit{DL-Lite}^{\mathcal{N}}_{bool}.$$

Similarly to the atemporal case, we can identify sub-Boolean fragments of the preceding languages. A temporal TBox is called a *Krom* or a *core* TBox if it contains only concept inclusions of the form

$$D_1 \sqsubseteq D_2, \qquad D_1 \sqsubseteq \neg D_2, \qquad \neg D_1 \sqsubseteq D_2, \qquad (\textit{krom})$$
$$D_1 \sqsubseteq D_2, \qquad D_1 \sqcap D_2 \sqsubseteq \bot, \qquad (\textit{core})$$

respectively, where the D_i are temporal concepts defined by (FPX), (FP), or (U) with

$$C \ ::= \ B \ \mid \ D.$$

Note that no Boolean operators are allowed in the D_i. This gives us six fragments: T_{FPX}*DL-Lite*$^{\mathcal{N}}_{\alpha}$, T_{FP}*DL-Lite*$^{\mathcal{N}}_{\alpha}$, and T_U*DL-Lite*$^{\mathcal{N}}_{\alpha}$, for $\alpha \in \{\textit{core}, \textit{krom}\}$.

Remark 3.1. We do not consider the core and Krom fragments of the full language with since ($\mathcal{S}$) and until ($\mathcal{U}$) because, as we shall see in Section 4.4 (Theorem 4.5), these operators allow one to go beyond the language of binary clauses of the core and Krom fragments, and the resulting languages would have the same complexity as $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ (but less expressive).

Remark 3.2. The introduced fragments of the full language $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ do not contain $\Diamond_F$ and $\Diamond_P$. Both operators, however, can be defined in the Krom and Bool fragments. For example, the concept inclusion $\Diamond_P B_1 \sqsubseteq \Diamond_F B_2$ can be represented by means of the inclusions

$$\Box_F A_2 \sqsubseteq \Box_P A_1 \qquad \text{and} \qquad A_i \sqsubseteq \neg B_i, \quad \neg B_i \sqsubseteq A_i, \qquad \text{for } i = 1, 2.$$

In the core fragments, where we do not have negation in the left-hand side, this trick does not work. Therefore, evolution constraints involving $\Diamond_P$ or $\Diamond_F$ (such as *Manager* $\sqsubseteq \Diamond_P$*Employee*) are not expressible in the core fragments (but timestamping remains expressible).

As we have seen in our running example, to express lifespan cardinality constraints, temporal operators on roles are required. For a role R of the form

$$R \ ::= \ S \ \mid \ S^- \ \mid \ \diamondplus R \ \mid \ \boxast R,$$

we define the extensions of $\diamondplus R$ and $\boxast R$ in an interpretation $\mathcal{I}$ by taking

$$(\diamondplus R)^{\mathcal{I}(n)} = \bigcup_{k \in \mathbb{Z}} R^{\mathcal{I}(k)} \quad \text{and} \quad (\boxast R)^{\mathcal{I}(n)} = \bigcap_{k \in \mathbb{Z}} R^{\mathcal{I}(k)}.$$

In this article, we consider three extensions of *DL-Lite*$^{\mathcal{N}}_{bool}$ with such temporalised roles, which are denoted by T^*_β*DL-Lite*$^{\mathcal{N}}_{bool}$, for $\beta \in \{X, FP, U\}$, where T^*_X*DL-Lite*$^{\mathcal{N}}_{bool}$ allows only $\bigcirc_P, \bigcirc_F$ as the temporal operators on concepts.

Table II. Complexity of the Temporal *DL-Lite* Logics

concept inclusions	temporal constructs		
	$\mathcal{U}/\mathcal{S}$, $\bigcirc_F/\bigcirc_P$, $\Box_F/\Box_P$[a]	$\Box_F/\Box_P$	$\boxast$
Bool	$T_{\mathcal{US}}DL\text{-}Lite^{\mathcal{N}}_{bool}$ $T_{FPX}DL\text{-}Lite^{\mathcal{N}}_{bool}$ PSPACE Thm. 4.4	$T_{FP}DL\text{-}Lite^{\mathcal{N}}_{bool}$ NP Thm. 4.6	$T_{U}DL\text{-}Lite^{\mathcal{N}}_{bool}$ NP Thm. 4.6
Krom	$T_{FPX}DL\text{-}Lite^{\mathcal{N}}_{krom}$ NP Thm. 4.7	$T_{FP}DL\text{-}Lite^{\mathcal{N}}_{krom}$ NP Thm. 4.7	$T_{U}DL\text{-}Lite^{\mathcal{N}}_{krom}$ NLOGSPACE Thm. 4.9
core	$T_{FPX}DL\text{-}Lite^{\mathcal{N}}_{core}$ NP Thm. 4.7	$T_{FP}DL\text{-}Lite^{\mathcal{N}}_{core}$ $\leq$ PTIME Thm. 4.8	$T_{U}DL\text{-}Lite^{\mathcal{N}}_{core}$ NLOGSPACE
temporalised roles	$T^{*}_{X}DL\text{-}Lite^{\mathcal{N}}_{bool}$ undec. Thm. 6.1	$T^{*}_{FP}DL\text{-}Lite^{\mathcal{N}}_{bool}$ undec. Thm. 6.1	$T^{*}_{U}DL\text{-}Lite^{\mathcal{N}}_{bool}$ NP Thm. 6.3
unrestricted role inclusions	$T_{\mathcal{US}}DL\text{-}Lite^{\mathcal{HN}}_{bool}$ undec. [Gabbay et al. 2003]	$T_{FP}DL\text{-}Lite^{\mathcal{HN}}_{bool}$ undec. [Gabbay et al. 2003]	$T^{*}_{U}DL\text{-}Lite^{\mathcal{HN}}_{bool}$ 2EXPTIME [Artale et al. 2007d]

[a]Sub-Boolean fragments of the language with $\mathcal{U}/\mathcal{S}$ are not define (see Remark 3.1).

We can also extend our languages with role inclusions, which are interpreted globally (in the same way as concept inclusions):

$$\mathcal{I} \models R_1 \sqsubseteq R_2 \quad \text{iff} \quad R_1^{\mathcal{I}(n)} \subseteq R_2^{\mathcal{I}(n)}, \qquad \text{for } all\ n \in \mathbb{Z}.$$

These extensions are denoted by $T_{\mathcal{US}}DL\text{-}Lite^{\mathcal{HN}}_{bool}$, $T_{FP}DL\text{-}Lite^{(\mathcal{HN})}_{bool}$, and so forth.

In the remaining part of the article, we investigate the computational complexity of the satisfiability problem for the temporal extensions of the *DL-Lite* logics designed earlier. But before that, we briefly summarise the obtained results in the more general context of TDLs.

3.3. Summary of the Complexity Results and Related Work

The temporal *DL-Lite* logics analysed here are collected in Table II together with the obtained and known complexity results. (Note that the complexity bounds in Table II are all tight except the case of $T_{FP}DL\text{-}Lite^{\mathcal{N}}_{core}$, where we only have an upper bound.) To avoid clutter, we omitted from the table the logics of the form $T_{\beta}DL\text{-}Lite^{(\mathcal{HN})}_{\alpha}$, whose complexity is the same as the complexity of the respective $T_{\beta}DL\text{-}Lite^{\mathcal{N}}_{\alpha}$.

The analysis of the constructs required for temporal conceptual modelling in Sections 2.1 and 3.1 has led us to temporalisations of *DL-Lite* logics, interpreted over the Cartesian products of object domains and the flow of time $(\mathbb{Z}, <)$, in which (1) the future and past temporal operators can be applied to concepts; (2) roles can be declared flexible or rigid; (3) the "undirected" temporal operators "always" and "sometime" can be applied to roles; (4) the concept and role inclusions are global, and the database (ABox) assertions are specified to hold at particular moments of time.

The minimal logic required to capture all of the temporal and static conceptual modelling constraints is $T^{*}_{FPX}DL\text{-}Lite^{\mathcal{HN}}_{bool}$; alas, it is undecidable. In fact, even the logic $T^{*}_{X}DL\text{-}Lite^{\mathcal{N}}_{bool}$, capturing only the quantitative evolution constraints, lifespan cardinalities, and covering, is undecidable. Replacing "quantitative" with "qualitative"—that is, considering $T^{*}_{FP}DL\text{-}Lite^{\mathcal{N}}_{bool}$—does not beat undecidability in the presence of lifespan cardinalities. Both of these undecidability results will still hold if we replace arbitrary cardinality constraints ($\mathcal{N}$) with role functionality. To regain decidability in the presence of temporalised roles, we have to limit the temporal operators on concepts to the undirected operators $\diamondast/\boxast$ —thus restricting the language to only timestamping and lifespan cardinalities. We show that the logic $T^{*}_{U}DL\text{-}Lite^{\mathcal{N}}_{bool}$ is NP-complete using the quasimodel technique.

Logics in the last row have arbitrary role inclusions, which together with functionality constraints are expressive enough to model all $\mathcal{ALC}$ constructors [Artale et al. 2007a, 2009a], and so the resulting TDLs are as complex as the corresponding temporal extensions of $\mathcal{ALC}$.

On a positive note, logics with restricted role inclusions and no temporal operators on roles exhibit much better computational properties. Our smallest logic, $T_U DL\text{-}Lite^{(\mathcal{HN})}_{core}$, is NLogSpace-complete. In the temporal dimension, it can only express timestamping constraints. It can also capture all of the static constraints that are different from covering and do not involve any interaction between role inclusions and number restrictions. Extending the language with covering leads to the loss of tractability in $T_U DL\text{-}Lite^{(\mathcal{HN})}_{bool}$. When covering is not needed and we are interested in temporal constraints different from lifespan cardinalities, we can regain tractability if we restrict the language to timestamping and evolution constraints that only capture persistence ($T_{FP} DL\text{-}Lite^{(\mathcal{HN})}_{core}$). If we also require migration constraints (that involve $\Diamond_P$ and $\Diamond_F$; see Remark 3.2) then we can use the Krom language $T_{FP} DL\text{-}Lite^{(\mathcal{HN})}_{krom}$, which is again NP-complete. Surprisingly, the addition of the full set of evolution constraints makes reasoning NP-complete even in $T_{FPX} DL\text{-}Lite^{(\mathcal{HN})}_{core}$.

To better appreciate the formalisms designed in this article, we consider them in a more general context of TDLs (for more detailed surveys, consult [Artale and Franconi 2001, 2005; Gabbay et al. 2003; Lutz et al. 2008]). Historically, the first temporal extensions of DLs were interval based [Schmiedel 1990]. Bettini [1997] considered interval-based temporal extensions of $\mathcal{ALC}$ in the style of Halpern and Shoham [1991] and established their undecidability. Artale and Franconi [1998] gave a subclass of decidable interval-based temporal DLs.

Numerous point-based temporal DLs have been constructed and investigated since the seminal paper of Schild [1993]. One of the lessons of the 20-year history of the discipline is that logics interpreted over 2D (or more) structures are very complex and sensitive to subtle interactions between constructs operating in different dimensions. The first TDLs suggested for representing TCMs were based on the expressive DLs $\mathcal{DLR}$ and $\mathcal{ALCQI}$ [Artale and Franconi 1999]. However, it turned out that already a single rigid role and the operator $\Diamond_F$ (or $\bigcirc_F$) on $\mathcal{ALC}$ concepts led to undecidability [Wolter and Zakharyaschev 1999]. In fact, to construct an undecidable TDL, one only needs a rigid role and three concept constructs: $\sqcap$, $\exists R.C$, and $\Diamond_F$—that is, a temporalised $\mathcal{EL}$ [Artale et al. 2007c]. There have been several attempts to tame the bad computational behaviour of TDLs by imposing various restrictions on the DL and temporal components as well as their interaction.

One approach was to disallow rigid roles and temporal operators on roles, which resulted in ExpSpace-complete temporalisations of $\mathcal{ALC}$ [Artale et al. 2002; Gabbay et al. 2003]. Such temporalisations reside in the monodic fragment[6] of first-order temporal logic [Hodkinson et al. 2000], for which tableau [Lutz et al. 2002; Kontchakov et al. 2004] and resolution [Degtyarev et al. 2006] reasoning algorithms have been developed and implemented [Hustadt et al. 2004; Guensel 2005; Ludwig and Hustadt 2010]. Another idea was to weaken the whole temporal component to the "undirected" temporal operators $\boxast$ and $\diamondast$ (which cannot discriminate between past, present, and future) on concepts and roles, resulting in a 2ExpTime-complete extension of $\mathcal{ALC}$ [Artale et al. 2007d]. The third approach was to allow arbitrary temporal operators on $\mathcal{ALC}$ axioms only (but not on concepts or roles) [Baader et al. 2008, 2012], which gave an ExpTime-complete logic. The addition of rigid concepts to this logic increases the complexity to NExpTime, whereas rigid concepts and roles make it 2ExpTime-complete. Finally, the

[6]A temporal formula is *monodic* if all of its subformulas beginning with a temporal operator have at most one free variable.

fourth approach, which dates back to Schild [1993], was to use only global axioms. In this case, $\mathcal{ALC}$ with temporal operators on concepts is ExpTime-complete, which matches the complexity of $\mathcal{ALC}$ itself (in contrast, temporal operators on axioms and concepts make the less expressive $\mathit{DL\text{-}Lite}_{bool}$ ExpSpace-complete [Artale et al. 2007c]).

As argued earlier, global axioms are precisely what we need in TCM. On the other hand, to capture timestamping and evolution constraints, we need the full set of temporal operators on concepts, whereas to capture lifespan cardinalities and timestamping on relations, we need temporalised or rigid roles. To achieve decidability in the case with rigid roles, we also weaken $\mathcal{ALC}$ to *DL-Lite*, which, as we have seen previously, perfectly suits the purpose of conceptual modelling. We thus start from the first promising results of Artale et al. [2009b], which demonstrated that even with rigid roles temporal extensions of $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ could be decidable, and extend them to various combinations of temporal operators and different sub-Boolean fragments of $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$, proving encouraging complexity results and showing how these logics can represent TCM schemas.

The results in the first three rows of Table II are established by using embeddings into the propositional temporal logic $\mathcal{PTL}$. To cope with the sub-Boolean core and Krom logics, we introduce, in Section 5, a number of new fragments of $\mathcal{PTL}$ by restricting the type of clauses in SNF [Fisher 1991] and the available temporal operators. The obtained complexity classification in Table III helps in understanding of the results in the first three rows of Table II.

4. REDUCING TEMPORAL *DL-Lite* TO PROPOSITIONAL TEMPORAL LOGIC

In this section, we reduce the satisfiability problem for $T_{\mathcal{US}}\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ KBs to the satisfiability problem for propositional temporal logic. This will be achieved in two steps. First, we reduce $T_{\mathcal{US}}\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ to the one-variable first-order temporal logic $\mathcal{QTL}^1$ [Gabbay et al. 2003]. Then we show that satisfiability of the resulting $\mathcal{QTL}^1$ formulas can be further reduced to satisfiability of $\mathcal{QTL}^1$ formulas without positive occurrences of existential quantifiers, which are essentially propositional temporal formulas. To simplify presentation, we consider here the logic $T_{\mathcal{US}}\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ (without role inclusions). The full $T_{\mathcal{US}}\mathit{DL\text{-}Lite}^{(\mathcal{HN})}_{bool}$ requires a bit more elaborate (yet absolutely routine) reduction that is similar to the one given by Artale et al. [2009b] for the atemporal case.

4.1. First-Order Temporal Logic

The language of first-order temporal logic $\mathcal{QTL}$ contains *predicates* $P_0, P_1, \dots$ (each with its arity), *variables* $x_0, x_1, \dots,$ and *constants* $a_0, a_1, \dots.$ *Formulas* φ of $\mathcal{QTL}$ are defined by the grammar:

$$\varphi \ ::= \ P_i(t_1, \dots, t_{k_i}) \ \mid \ \bot \ \mid \ \forall x\, \varphi \ \mid \ \neg\varphi \ \mid \ \varphi_1 \land \varphi_2 \ \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \ \mid \ \varphi_1 \,\mathcal{S}\, \varphi_2,$$

where k_i is the arity of P_i and the t_j are *terms*-that is, variables or constants. These formulas are interpreted in first-order temporal models $\mathfrak{M}$, which, for every $n \in \mathbb{Z}$, give a first-order structure

$$\mathfrak{M}(n) = (\Delta^{\mathfrak{M}}, a_0^{\mathfrak{M}}, a_1^{\mathfrak{M}}, \dots, P_0^{\mathfrak{M},n}, P_1^{\mathfrak{M},n} \dots)$$

with the same domain $\Delta^{\mathfrak{M}}$, the same $a_i^{\mathfrak{M}} \in \Delta$, for each constant a_i, and where $P_i^{\mathfrak{M},n}$ is a k_i-ary relation on $\Delta^{\mathfrak{M}}$, for each predicate P_i of arity k_i and each $n \in \mathbb{Z}$. An *assignment* in $\mathfrak{M}$ is a function, $\mathfrak{a}$, that maps variables to elements of $\Delta^{\mathfrak{M}}$. For a term t, we write $t^{\mathfrak{a},\mathfrak{M}}$ for $\mathfrak{a}(x)$ if $t = x$, and for $a^{\mathfrak{M}}$ if $t = a$. The semantics of $\mathcal{QTL}$ is standard (e.g., see

Gabbay et al. [2003]):

$$
\begin{array}{l}
\mathfrak{M}, n \models^{\mathfrak{a}} P(t_1, \dots, t_k) \quad \text{iff} \quad (t_1^{\mathfrak{a},\mathfrak{M}}, \dots, t_k^{\mathfrak{a},\mathfrak{M}}) \in P^{\mathfrak{M},n},\\
\mathfrak{M}, n \not\models^{\mathfrak{a}} \bot,\\
\mathfrak{M}, n \models^{\mathfrak{a}} \forall x\, \varphi \quad \text{iff} \quad \mathfrak{M}, n \models^{\mathfrak{a}'} \varphi, \text{ for all assignments } \mathfrak{a}' \text{ that differ from } \mathfrak{a} \text{ on } x \text{ only},\\
\mathfrak{M}, n \models^{\mathfrak{a}} \neg\varphi \quad \text{iff} \quad \mathfrak{M}, n \not\models^{\mathfrak{a}} \varphi,\\
\mathfrak{M}, n \models^{\mathfrak{a}} \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad \mathfrak{M}, n \models^{\mathfrak{a}} \varphi_1 \text{ and } \mathfrak{M}, n \models^{\mathfrak{a}} \varphi_2,\\
\mathfrak{M}, n \models^{\mathfrak{a}} \varphi_1 \,\mathcal{U}\, \varphi_2 \quad \text{iff} \quad \text{there is } k > n \text{ with } \mathfrak{M}, k \models^{\mathfrak{a}} \varphi_2 \text{ and } \mathfrak{M}, m \models^{\mathfrak{a}} \varphi_1, \text{ for } n < m < k,\\
\mathfrak{M}, n \models^{\mathfrak{a}} \varphi_1 \,\mathcal{S}\, \varphi_2 \quad \text{iff} \quad \text{there is } k < n \text{ with } \mathfrak{M}, k \models^{\mathfrak{a}} \varphi_2 \text{ and } \mathfrak{M}, m \models^{\mathfrak{a}} \varphi_1, \text{ for } k < m < n.
\end{array}
$$

We use the standard abbreviations such as

$$
\top = \neg\bot, \qquad \varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2), \qquad \exists x\, \varphi = \neg\forall x\, \neg\varphi,
$$
$$
\Diamond_F \varphi = \top \,\mathcal{U}\, \varphi, \qquad \Box_F \varphi = \neg\Diamond_F\neg\varphi, \qquad \bigcirc_F \varphi = \bot \,\mathcal{U}\, \varphi
$$

as well as the past counterparts for $\Diamond_P$, $\Box_P$, and $\bigcirc_P$; we also write $\boxast\, \varphi$ for $\Box_F \Box_P\, \varphi$.

If a formula φ contains no free variables (i.e., φ is a sentence), then we omit the valuation $\mathfrak{a}$ in $\mathfrak{M}, n \models^{\mathfrak{a}} \varphi$ and write $\mathfrak{M}, n \models \varphi$. If φ has a single free variable x, then we write $\mathfrak{M}, n \models \varphi[a]$ in place of $\mathfrak{M}, n \models^{\mathfrak{a}} \varphi$ with $\mathfrak{a}(x) = a$.

A $\mathcal{QTL}^1$*-formula* is a $\mathcal{QTL}$-formula that is constructed using at most one variable. Satisfiability of $\mathcal{QTL}^1$-formulas is known to be ExpSpace-complete [Gabbay et al. 2003]. In the propositional temporal logic, $\mathcal{PTL}$, only 0-ary predicates (i.e., propositional variables) are allowed. The satisfiability problem for $\mathcal{PTL}$-formulas is PSpace-complete [Sistla and Clarke 1982].

4.2. Reduction to $\mathcal{QTL}^1$

Given a $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, let $\mathsf{ob}_{\mathcal{A}}$ be the set of all object names occurring in $\mathcal{A}$ and $\mathsf{role}_{\mathcal{K}}$ the set of rigid and flexible role names occurring in $\mathcal{K}$ and their inverses.

In our reduction, objects $a \in \mathsf{ob}_{\mathcal{A}}$ are mapped to constants a, concept names A to unary predicates $A(x)$, and number restrictions $\geq q\, R$ to unary predicates $E_q R(x)$. Intuitively, for a role name S, the predicates $E_q S(x)$ and $E_q S^-(x)$ represent, at each moment of time, the sets of elements with *at least* q distinct S-successors and *at least* q distinct S-predecessors; in particular, $E_1 S(x)$ can be thought of as the domain of S and $E_1 S^-(x)$ as the range of S. By induction on the construction of a $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ concept C, we define the $\mathcal{QTL}^1$-formula $C^*(x)$:

$$
A^* = A(x), \qquad \bot^* = \bot, \qquad (\geq q\, R)^* = E_q R(x),
$$
$$
(C_1 \,\mathcal{U}\, C_2)^* = C_1^* \,\mathcal{U}\, C_2^*, \quad (C_1 \,\mathcal{S}\, C_2)^* = C_1^* \,\mathcal{S}\, C_2^*, \quad (C_1 \sqcap C_2)^* = C_1^* \wedge C_2^*, \quad (\neg C)^* = \neg C^*.
$$

It can be easily seen that the map $\cdot^*$ commutes with all the Boolean and temporal operators—for example, $(\Diamond_F C)^* = \Diamond_F C^*$. For a TBox $\mathcal{T}$, we consider the following sentence, saying that the concept inclusions in $\mathcal{T}$ hold globally:

$$
\mathcal{T}^\dagger = \bigwedge_{C_1 \sqsubseteq C_2 \in \mathcal{T}} \boxast\, \forall x\, \big(C_1^*(x) \to C_2^*(x)\big).
$$

In the preceding translation, we replaced binary predicates (i.e., roles) by collections of unary predicates, the $E_q R(x)$. Clearly, we have to ensure that these predicates behave similarly to the respective number restrictions. In particular, the following three properties trivially hold in $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ interpretations, for all roles R at all moments of time:

—Every point with at least q' R-successors has at least q R-successors, for each $q < q'$.
—If R is a rigid role, then every point with at least q R-successors at some moment has at least q R-successors at all moments of time.
—If the domain of a role is not empty, then its range is not empty either.

These conditions can be encoded by the following $\mathcal{QTL}^1$-sentences:

$$\bigwedge_{R \in \mathsf{role}_{\mathcal{K}}} \quad \bigwedge_{q,q' \in Q_{\mathcal{T}} \text{ with } q' > q} \boxast\, \forall x\, \big((\geq q'\, R)^*(x) \to (\geq q\, R)^*(x)\big), \tag{5}$$

$$\bigwedge_{R \in \mathsf{role}_{\mathcal{K}} \text{ is rigid}} \quad \bigwedge_{q \in Q_{\mathcal{T}}} \boxast\, \forall x\, \big((\geq q\, R)^*(x) \to \boxast\, (\geq q\, R)^*(x)\big), \tag{6}$$

$$\bigwedge_{R \in \mathsf{role}_{\mathcal{K}}} \boxast\, \big(\exists x\, (\exists R)^*(x) \;\to\; \exists x\, (\exists \mathsf{inv}(R))^*(x)\big), \tag{7}$$

where $Q_{\mathcal{T}}$ is the set containing 1 and all numbers q such that $\geq q\, R$ occurs in $\mathcal{T}$, and $\mathsf{inv}(R)$ is the inverse of R, for instance, $\mathsf{inv}(S) = S^-$ and $\mathsf{inv}(S^-) = S$, for a role name S. As we shall see later, these three properties are enough to ensure that the real binary relations for roles S in $T_{\mathcal{US}}DL\text{-}Lite^{\mathcal{N}}_{bool}$ can be reconstructed from the collections of unary predicates $E_q S(x)$ and $E_q S^-(x)$ satisfying (5) through (7).

It is easy to extend the preceding reduction to ABox concept assertions: take $\bigcirc^n A(a)$ for each $\bigcirc^n A(a) \in \mathcal{A}$, and $\bigcirc^n \neg A(a)$ for each $\bigcirc^n \neg A(a) \in \mathcal{A}$. However, ABox role assertions need a more elaborate treatment. For every $a \in \mathsf{ob}_{\mathcal{A}}$, if a has q R-successors in $\mathcal{A}$ at moment n-that is, $\bigcirc^n R(a, b_1), \dots, \bigcirc^n R(a, b_q) \in \mathcal{A}$, for distinct $b_1, \dots, b_q$—then we include $(\bigcirc^n \geq q\, R)^*(a)$ in the translation of $\mathcal{A}$. When counting the number of successors, one has to remember the following property of rigid roles S: if an ABox contains $\bigcirc^m S(a, b)$, then $\bigcirc^n S(a, b)$ holds for all $n \in \mathbb{Z}$, and so $\bigcirc^m S(a, b)$ contributes to the number of S-successors of a and S^--successors of b at *each* moment.

In what follows, we assume that $\mathcal{A}$ contains $\bigcirc^n S^-(b, a)$ whenever it contains $\bigcirc^n S(a, b)$. For each $n \in \mathbb{Z}$ and each role R, we define the temporal slice $\mathcal{A}^R_n$ of $\mathcal{A}$ by taking

$$\mathcal{A}^R_n = \begin{cases} \big\{R(a, b) \mid \bigcirc^m R(a, b) \in \mathcal{A} \text{ for some } m \in \mathbb{Z}\big\}, & R \text{ is a rigid role,} \\ \big\{R(a, b) \mid \bigcirc^n R(a, b) \in \mathcal{A}\big\}, & R \text{ is a flexible role.} \end{cases}$$

The translation $\mathcal{A}^\dagger$ of the $T_{\mathcal{US}}DL\text{-}Lite^{\mathcal{N}}_{bool}$ ABox $\mathcal{A}$ is defined now by taking

$$\mathcal{A}^\dagger = \bigwedge_{\bigcirc^n A(a) \in \mathcal{A}} \bigcirc^n A(a) \;\wedge \bigwedge_{\bigcirc^n \neg A(a) \in \mathcal{A}} \bigcirc^n \neg A(a) \;\wedge \bigwedge_{\bigcirc^n R(a,b) \in \mathcal{A}} \bigcirc^n \big(\geq q^{R,n}_{\mathcal{A}(a)} R\big)^*(a) \;\wedge \bigwedge_{\substack{\bigcirc^n \neg S(a,b) \in \mathcal{A} \\ S(a,b) \in \mathcal{A}^S_n}} \bot,$$

where $q^{R,n}_{\mathcal{A}(a)}$ is the number of distinct R-successors of a in $\mathcal{A}$ at moment n:

$$q^{R,n}_{\mathcal{A}(a)} = \max\big\{q \in Q_{\mathcal{T}} \mid R(a, b_1), \dots, R(a, b_q) \in \mathcal{A}^R_n, \text{ for distinct } b_1, \dots, b_q\big\}.$$

We note that $\mathcal{A}^\dagger$ can be effectively computed for any given $\mathcal{K}$ because we need temporal slices $\mathcal{A}^R_n$ only for those n that are explicitly mentioned in $\mathcal{A}$—that is, those n with $\bigcirc^n R(a, b) \in \mathcal{A}$.

Finally, we define the $\mathcal{QTL}^1$-*translation* $\mathcal{K}^\dagger$ of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ as the conjunction of $\mathcal{T}^\dagger$, $\mathcal{A}^\dagger$, and formulas (5) through (7). The size of $\mathcal{T}^\dagger$ and $\mathcal{A}^\dagger$ does not exceed the size of $\mathcal{T}$ and $\mathcal{A}$, respectively. The size of (6) and (7) is linear in the size of $\mathcal{T}$, whereas the size of (5) is cubic in the size of $\mathcal{T}$ (although it can be made linear by taking account of only those q that occur in $\geq q\, R$, for a fixed R, and replacing $q' > q$ in the conjunction index with a more restrictive condition "$q' > q$ and there is no $q'' \in Q_{\mathcal{T}}$ with $q' > q'' > q$"; for details, see Artale et al. [2009a]).

The main technical result of this section is that $\mathcal{K}$ and $\mathcal{K}^{\dagger}$ are equisatisfiable; the proof (based on the unravelling construction) is given in Appendix A.

THEOREM 4.1. *A $T_{\mathcal{US}}$DL-Lite$^{\mathcal{N}}_{bool}$ KB $\mathcal{K}$ is satisfiable iff the $\mathcal{QTL}^1$-sentence $\mathcal{K}^{\dagger}$ is satisfiable.*

Meanwhile, we proceed to the second step of our reduction.

4.3. Reduction to $\mathcal{PTL}$

Our next aim is to construct a $\mathcal{PTL}$-formula that is equisatisfiable with $\mathcal{K}^{\dagger}$. First, we observe that $\mathcal{K}^{\dagger}$ can be represented in the form $\mathcal{K}^{\dagger_0} \wedge \bigwedge_{R \in \mathsf{role}_{\mathcal{K}}} \vartheta_R$, where

$$\mathcal{K}^{\dagger_0} = \boxast \forall x\, \varphi(x) \;\wedge\; \psi \qquad \text{and} \qquad \vartheta_R = \boxast \forall x\, \big((\exists R)^*(x) \to \exists x\, (\exists \mathsf{inv}(R))^*(x)\big),$$

for a quantifier-free first-order temporal formula $\varphi(x)$ with a single variable x and unary predicates only and a variable-free formula ψ. We show now that one can replace ϑ_R by a formula without existential quantifiers. To this end, we require the following lemma:

LEMMA 4.2. *For every $T_{\mathcal{US}}$DL-Lite$^{\mathcal{N}}_{bool}$ KB $\mathcal{K}$, if there is a model $\mathfrak{M}$ satisfying $\mathcal{K}^{\dagger_0}$ such that $\mathfrak{M}, n_0 \models (\exists R)^*[d]$, for some $n_0 \in \mathbb{Z}$ and $d \in \Delta^{\mathfrak{M}}$, then there is a model $\mathfrak{M}'$ extending $\mathfrak{M}$ with new elements and satisfying $\mathcal{K}^{\dagger_0}$ such that for each $n \in \mathbb{Z}$, there is $d_n \in \Delta^{\mathfrak{M}'}$ with $\mathfrak{M}', n \models (\exists R)^*[d_n]$.*

PROOF. Consider a new model $\mathfrak{M}'$ with domain $\Delta^{\mathfrak{M}} \cup (\{d\} \times \mathbb{Z})$ by setting

$$B^{\mathfrak{M}',n} = B^{\mathfrak{M},n} \cup \big\{(d,k) \mid d \in B^{\mathfrak{M},n-k}, k \in \mathbb{Z}\big\}, \qquad \text{for each } B \text{ of } \mathcal{K}^{\dagger_0} \text{ and } n \in \mathbb{Z}.$$

In other words, $\mathfrak{M}'$ is the disjoint union of $\mathfrak{M}$ and copies of d "shifted" along the timeline by k steps, for each $k \in \mathbb{Z}$. It follows that at each moment $n \in \mathbb{Z}$, the element $(d, n - n_0)$ belongs to $(\exists R)^*$, thus making $(\exists R)^*$ nonempty at all moments of time. Moreover, $\mathfrak{M}', 0 \models \mathcal{K}^{\dagger_0}$ because $\varphi(x)$ expresses a property of a single domain element and holds at *each* moment of time, ψ depends only on the part of the model that corresponds to constants (and which are interpreted as in $\mathfrak{M}$). □

Next, for each $R \in \mathsf{role}_{\mathcal{K}}$, we take a fresh constant d_R and a fresh propositional variable p_R (recall that $\mathsf{inv}(R)$ is also in $\mathsf{role}_{\mathcal{K}}$), and consider the following $\mathcal{QTL}^1$-formula:

$$\mathcal{K}^{\ddagger} = \mathcal{K}^{\dagger_0} \wedge \bigwedge\nolimits_{R \in \mathsf{role}_{\mathcal{K}}} \vartheta'_R, \text{ where } \vartheta'_R \;=\; \boxast \forall x\, \big((\exists R)^*(x) \to \boxast\, p_R\big) \wedge \big(p_{\mathsf{inv}(R)} \to (\exists R)^*(d_R)\big)$$

($p_{\mathsf{inv}(R)}$ and p_R indicate that $\mathsf{inv}(R)$ and R are nonempty, whereas d_R and $d_{\mathsf{inv}(R)}$ witness that at 0).

LEMMA 4.3. *A $T_{\mathcal{US}}$DL-Lite$^{\mathcal{N}}_{bool}$ KB $\mathcal{K}$ is satisfiable iff the $\mathcal{QTL}^1$-sentence $\mathcal{K}^{\ddagger}$ is satisfiable.*

PROOF. ($\Rightarrow$) If $\mathcal{K}$ is satisfiable, then by Theorem 4.1 and repeated application of Lemma 4.2, $\mathcal{K}^{\dagger_0}$ is satisfied in a model $\mathfrak{M}$ such that for each $R \in \mathsf{role}_{\mathcal{K}}$, the predicates $(\exists R)^*$ and $(\exists \mathsf{inv}(R))^*$ are either both empty at all moments of time or both nonempty at all moments of time. To satisfy the ϑ'_R, for $R \in \mathsf{role}_{\mathcal{K}}$, we extend $\mathfrak{M}$ to $\mathfrak{M}'$ as follows: if $(\exists R)^*$ and $(\exists \mathsf{inv}(R))^*$ are nonempty, we set $p_R^{\mathfrak{M}',n}$ to be true at all $n \in \mathbb{Z}$ and take d_R to be an element in $((\exists R)^*)^{\mathfrak{M},0}$; otherwise, we set $p_R^{\mathfrak{M}',0}$ to be false and take some domain element as d_R. It follows that $\mathfrak{M}', 0 \models \mathcal{K}^{\ddagger}$.

($\Leftarrow$) Conversely, suppose that $\mathfrak{M}, 0 \models \mathcal{K}^{\ddagger}$. By repeated application of Lemma 4.2, $\mathcal{K}^{\dagger_0}$ is satisfied in a model $\mathfrak{M}'$ such that for each $R \in \mathsf{role}_{\mathcal{K}}$, the predicates $(\exists R)^*$ and $(\exists \mathsf{inv}(R))^*$ are either both empty at all moments of time or both nonempty at all moments of time. It follows that $\mathfrak{M}', 0 \models \vartheta_R$ for all $R \in \mathsf{role}_{\mathcal{K}}$, and so $\mathfrak{M}', 0 \models \mathcal{K}^{\dagger}$. □

Finally, as $\mathcal{K}^{\ddagger}$ contains no existential quantifiers, it can be regarded as a propositional temporal formula because all universally quantified variables can be instantiated by all constants in the formula (which only results in a polynomial blowup). Observe also that the translation $\cdot^{\ddagger}$ can be done in logarithmic space in the size of $\mathcal{K}$. This is almost trivial for all conjuncts of $\mathcal{K}^{\ddagger}$ apart from $(\geq q^{R,n}_{\mathcal{A}(a)} R)^{*}$ in $\mathcal{A}^{\dagger}$, where the numbers can be computed using a LogSpace-transducer as follows: initially set $q = 0$; then enumerate all object names b_i in $\mathcal{A}$ incrementing q for each $R(a, b_i) \in \mathcal{A}^n_R$ and stop if $q = \max \mathcal{Q}_{\mathcal{T}}$ or the end of the object names list is reached; let $q^{R,n}_{\mathcal{A}(a)}$ be the maximum number in $\mathcal{Q}_{\mathcal{T}}$ not exceeding q. Note that in the case of $T_{\mathcal{US}}$*DL-Lite*$^{(\mathcal{HN})}_{bool}$, the translation is feasible only in NLogSpace (rather than LogSpace) because we have to take account of role inclusions (and graph reachability is NLogSpace-complete).

4.4. Complexity of $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ and Its Fragments

We now use the translation $\cdot^{\ddagger}$ to obtain the complexity results announced in Section 3.3.

Theorem 4.4. *The satisfiability problem for* $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ *and* T_{FPX}*DL-Lite*$^{\mathcal{N}}_{bool}$ *KBs is* PSpace-*complete.*

Proof. The upper bound follows from the reduction $\cdot^{\ddagger}$ above and the fact that $\mathcal{PTL}$ is PSpace-complete over $(\mathbb{Z}, <)$ [Rabinovich 2010; Reynolds 2010; Sistla and Clarke 1982]. The lower bound is an immediate consequence of the observation that T_{FPX}*DL-Lite*$^{\mathcal{N}}_{bool}$ can encode formulas of the form $\theta \wedge \boxed{*} \bigwedge_i (\varphi_i \rightarrow \bigcirc_F \psi_i)$, where θ, the φ_i and ψ_i are conjunctions of propositional variables: satisfiability of such formulas is known to be PSpace-hard (see e.g., Gabbay et al. [1994]). □

In fact, using the $\mathcal{U}$-operator, we can establish the following:

Theorem 4.5. *The satisfiability problem for the core fragment of* $T_{\mathcal{US}}$*DL-Lite*$^{\mathcal{N}}_{bool}$ *KBs is* PSpace-*hard.*

Proof. The proof is by reduction of the nonhalting problem for deterministic Turing machines with a polynomial tape. Let $s(n)$ be a polynomial and M a deterministic Turing machine that requires $s(n)$ cells of the tape given an input of length n. Without loss of generality, we assume that M never runs outside the first $s(n)$ cells. Let $M = \langle Q, \Gamma, \#, \Sigma, \delta, q_0, q_f\rangle$, where Q is a finite set of states, Γ a tape alphabet, $\# \in \Gamma$ the blank symbol, $\Sigma \subseteq \Gamma$ a set of input symbols, $\delta : (Q \setminus \{q_f\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ a transition function, and $q_0, q_f \in Q$ the initial and accepting states, respectively. Let $\vec{a} = a_1 \dots a_n$ be an input for M. We construct a KB that is unsatisfiable iff M accepts $\vec{a}$. This will prove PSpace-hardness. The KB uses a single object name d and the following concepts, for $a \in \Gamma$, $q \in Q$, and $1 \leq i \leq s(n)$, representing configurations of M:

—H_{iq}, which contains d if the head points to cell i and the current state is q;
—S_{ia}, which contains d if tape cell i contains symbol a in the current configuration;
—D_i, which contains d if the head pointed to cell i in the *previous* configuration.

Let $\mathcal{T}_M$ contain the following concept inclusions, for $a, a' \in \Gamma$, $q, q' \in Q$, $1 \leq i, j \leq s(n)$:

$$H_{iq} \sqsubseteq \bot\, \mathcal{U}\, H_{(i+1)q'}, \quad H_{iq} \sqsubseteq \bot\, \mathcal{U}\, S_{ia'}, \qquad \text{if } \delta(q, a) = (q', a', R) \text{ and } i < s(n), \tag{8}$$

$$H_{iq} \sqsubseteq \bot\, \mathcal{U}\, H_{(i-1)q'}, \quad H_{iq} \sqsubseteq \bot\, \mathcal{U}\, S_{ia'}, \qquad \text{if } \delta(q, a) = (q', a', L) \text{ and } i > 1, \tag{9}$$

$$H_{iq} \sqsubseteq \bot\, \mathcal{U}\, D_i, \tag{10}$$

$$D_i \sqcap D_j \sqsubseteq \bot, \qquad \text{if } i \neq j, \tag{11}$$

$$S_{ia} \sqsubseteq S_{ia}\, \mathcal{U}\, D_i, \tag{12}$$

$$H_{iq_f} \sqsubseteq \bot, \tag{13}$$

and let $\mathcal{A}_{\vec{a}}$ consist of the following ABox assertions:

$$H_{1q_0}(d), \qquad S_{ia_i}(d), \text{ for } 1 \leq i \leq n, \qquad S_{i\#}(d), \text{ for } n < i \leq s(n).$$

Note that the concept inclusions in $\mathcal{T}_M$ are of the form $B_1 \sqcap B_2 \sqsubseteq \bot$ or $B_1 \sqsubseteq B_2 \,\mathcal{U}\, B_3$, where each B_i is either a concept name or $\bot$, and that $\mathcal{U}$, in essence, encodes the "next-time" operator. The proof that $(\mathcal{T}_M, \mathcal{A}_{\vec{a}})$ is unsatisfiable iff M accepts $\vec{a}$ is given in Appendix B. $\square$

On the other hand, if we do not have the $\mathcal{U}/\mathcal{S}$ or $\bigcirc_F/\bigcirc_P$ operators in our languages, then the complexity drops to NP, which matches the complexity of the $\Box_F/\Box_P$-fragment of propositional temporal logic [Ono and Nakamura 1980]:

THEOREM 4.6. *Satisfiability of T_{FP}DL-Lite$^{\mathcal{N}}_{bool}$ and T_UDL-Lite$^{\mathcal{N}}_{bool}$ KBs is* NP*-complete.*

PROOF. The lower bound is immediate from the complexity of *DL-Lite*$^{\mathcal{N}}_{bool}$. The upper bound for T_{FP}*DL-Lite*$^{\mathcal{N}}_{bool}$ can be shown using a slight modification of the reduction $\cdot^{\ddagger}$ and the result of Ono and Nakamura [1980] mentioned earlier. We need to modify $\cdot^{\ddagger}$ in such a way that the target language does not contain the $\bigcirc^n$ operators of the ABox. We take a fresh predicate $H^n_C(x)$ for each ground atom $\bigcirc^n C(a)$ occurring in $\mathcal{A}^{\dagger}$ and use the following formulas instead of $\bigcirc^n C(a)$ in $\mathcal{A}^{\dagger}$:

$$\big(\Diamond^n_F H^n_C(a) \wedge \neg\Diamond^{n+1}_F H^n_C(a)\big) \wedge \boxast \; \big(H^n_C(a) \to C(a)\big), \qquad \text{if } n \geq 0,$$
$$\big(\Diamond^{-n}_P H^n_C(a) \wedge \neg\Diamond^{-n+1}_P H^n_C(a)\big) \wedge \boxast \; \big(H^n_C(a) \to C(a)\big), \qquad \text{if } n < 0,$$

where $\Diamond^n_F$ and $\Diamond^n_P$ denote n applications of $\Diamond_F$ and $\Diamond_P$, respectively. Note that $H^n_C(a)$ holds at m iff $m = n$. Thus, we use these predicates to "mark" a small number of moments of time in models.

The NP upper bound trivially holds for T_U*DL-Lite*$^{\mathcal{N}}_{bool}$, a sublanguage of T_{FP}*DL-Lite*$^{\mathcal{N}}_{bool}$. $\square$

Our next theorem also uses the reduction $\cdot^{\ddagger}$ and follows from the complexity results for the fragments $\mathcal{PTL}_{krom}(\boxast, \bigcirc_F/\bigcirc_P, \Box_F/\Box_P)$ and $\mathcal{PTL}_{core}(\boxast, \bigcirc_F/\bigcirc_P)$ of $\mathcal{PTL}$, obtained by restricting the form of clauses in the SNF [Fisher 1991] and proved in Section 5.

THEOREM 4.7. *Satisfiability of T_{FPX}DL-Lite$^{\mathcal{N}}_{krom}$, T_{FP}DL-Lite$^{\mathcal{N}}_{krom}$, and T_{FPX}DL-Lite$^{\mathcal{N}}_{core}$ KBs is* NP*-complete.*

PROOF. The NP upper bound follows from the fact that the $\cdot^{\ddagger}$ translation of a KB in any of the three languages is a $\mathcal{PTL}_{krom}(\boxast, \bigcirc_F/\bigcirc_P, \Box_F/\Box_P)$-formula. By Theorem 5.4, satisfiability of such formulas is in NP. The matching lower bound for T_{FP}*DL-Lite*$^{\mathcal{N}}_{krom}$ (and T_{FPX}*DL-Lite*$^{\mathcal{N}}_{krom}$) follows from the proof of NP-hardness of $\mathcal{PTL}_{krom}(\boxast, \Box_P/\Box_F)$, which will be presented in Theorem 5.8: one can take concept names instead of propositional variables and encode, in an obvious way, the formulas of the proof of Theorem 5.8 in a KB; similarly, the lower bound for T_{FPX}*DL-Lite*$^{\mathcal{N}}_{core}$ follows from NP-hardness of $\mathcal{PTL}_{core}(\boxast, \bigcirc_F/\bigcirc_P)$; see Theorem 5.4. $\square$

THEOREM 4.8. *Satisfiability of T_{FP}DL-Lite$^{\mathcal{N}}_{core}$ KBs is in* PTIME.

PROOF. The result follows from the observation that the $\cdot^{\ddagger}$ translation of a T_{FP}*DL-Lite*$^{\mathcal{N}}_{core}$ KB is of the form $\varphi' \wedge \varphi''$, where φ' is a $\mathcal{PTL}_{core}(\boxast, \Box_F/\Box_P)$-formula representing the TBox and φ'' is a conjunction of formulas of the form $\bigcirc^n p$, for propositional variables p. A modification of the proof of Theorem 5.5 (explained in Remark 5.6) shows that satisfiability of such formulas is in PTIME.

We note in passing that the matching lower bound for $\mathcal{PTL}_{core}(\boxast, \Box_F/\Box_P)$, to be proved in Theorem 5.7, does not imply PTIME-hardness of T_{FP}*DL-Lite*$^{\mathcal{N}}_{core}$, as the formulas in the proof require an implication to hold at the initial moment of time, which is not expressible in T_{FP}*DL-Lite*$^{\mathcal{N}}_{core}$. $\square$

Finally, we show that the Krom and core fragments of T_U*DL-Lite*$^{\mathcal{N}}_{bool}$ can be simulated by 2CNFs (e.g., see Börger et al. [1997]), whose satisfiability is NLogSpace-complete.

Theorem 4.9. *The satisfiability problem for* T_U*DL-Lite*$^{\mathcal{N}}_{core}$ *and* T_U*DL-Lite*$^{\mathcal{N}}_{krom}$ *KBs is* NLogSpace-*complete.*

Proof. The lower bound is trivial from NLogSpace-hardness of *DL-Lite*$^{\mathcal{N}}_{core}$. We show the matching upper bound. Given a T_U*DL-Lite*$^{\mathcal{N}}_{krom}$ KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, we consider the $\mathcal{QTL}^1$-formula $\mathcal{K}^\ddagger$, which, by Lemma 4.3, is satisfiable iff $\mathcal{K}$ is satisfiable. Now, we transform $\mathcal{K}^\ddagger$ into a two-sorted first-order formula $\mathcal{K}^{\ddagger_2}$ by representing the time dimension explicitly as a predicate argument. Recall that $\mathcal{K}^\ddagger$ is built from the propositional variables p_R for $R \in \mathsf{role}_\mathcal{K}$, and unary predicates B^* for concepts B of the form A and $\geq q\, R$. Without loss of generality, we assume that there is at most one $\boxast$ in front of each B^* in $\mathcal{K}^\ddagger$. We replace each $B^*(x)$ in $\mathcal{K}^\ddagger$ that is not prefixed by $\boxast$ with the binary predicate $B^*(x,t)$, and each $\boxast\, B^*(x)$ with a fresh unary predicate $U_B(x)$; the outermost $\boxast$ is replaced by $\forall t$. To preserve the semantics of the $\boxast\, B^*$, we also append to the resulting formula the conjuncts $\forall x\, (U_B(x) \leftrightarrow \forall t\, B^*(x,t))$, which are equivalent to

$$\forall t\, \forall x\, \big(U_B(x) \to B^*(x,t)\big) \quad \wedge \quad \forall x\, \exists t\, \big(B^*(x,t) \to U_B(x)\big).$$

The propositional variables p_R of $\mathcal{K}^\ddagger$ remain propositional variables in $\mathcal{K}^{\ddagger_2}$, and the second conjunct of $\mathcal{K}^\ddagger$ is replaced by the following formula:

$$\big(p_R \to (\exists \mathsf{inv}(R))^*(d_{\mathsf{inv}(R)}, 0)\big) \quad \wedge \quad \forall t\, \forall x\, ((\exists R)^*(x,t) \to p_R)$$

with constant 0. Finally, the ground atoms $\bigcirc^n B^*(a)$ in $\mathcal{A}^\dagger$ are replaced by $B^*(a,n)$ with constants n. Thus, $\mathcal{K}^{\ddagger_2}$ is a conjunction of (at most) binary clauses without quantifiers or with prefixes of the form $\forall t\, \forall x$ and $\forall x\, \exists t$. Since the first argument of the predicates, x, is always universally quantified, $\mathcal{K}^{\ddagger_2}$ is equisatisfiable with the conjunction $\mathcal{K}^{\ddagger_3}$ of the formulas obtained by replacing x in $\mathcal{K}^{\ddagger_2}$ with the constants in the set $\mathsf{ob}_\mathcal{A} \cup \{d_R \mid R \in \mathsf{role}_\mathcal{K}\}$. But then $\mathcal{K}^{\ddagger_3}$ is equivalent to a first-order Krom formula in prenex form with the quantifier prefix $\exists^*\forall$, satisfiability of which can be checked in NLogSpace (e.g., see Theorem 8.3.6 in Börger et al. [1997]). $\square$

5. CLAUSAL FRAGMENTS OF PROPOSITIONAL TEMPORAL LOGIC

Our aim in this section is to introduce and investigate a number of new fragments of the propositional temporal logic $\mathcal{PTL}$. One reason for this is to obtain the complexity results required for the proof of Theorems 4.7 and 4.8. We believe, however, that these fragments are of sufficient interest on their own, independently of temporal conceptual modelling and reasoning.

Sistla and Clarke [1982] showed that full $\mathcal{PTL}$ is PSpace-complete (see also Halpern and Reif [1981], Lichtenstein et al. [1985], Rabinovich [2010], and Reynolds [2010]). Ono and Nakamura [1980] proved that for formulas with only $\Box_F$ and $\Diamond_F$, the satisfiability problem becomes NP-complete. Since then a number of fragments of $\mathcal{PTL}$ with lower computational complexity have been identified and studied. Chen and Lin [1993] observed that the complexity of $\mathcal{PTL}$ does not change even if we restrict attention to temporal Horn formulas. Demri and Schnoebelen [2002] determined the complexity of fragments that depend on three parameters: the available temporal operators, the number of nested temporal operators, and the number of propositional variables in formulas. Markey [2004] analysed fragments defined by the allowed set of temporal operators, their nesting, and the use of negation. Dixon et al. [2007] introduced a XOR fragment of $\mathcal{PTL}$ and showed its tractability. Bauland et al. [2009] systematically investigated the complexity of fragments given by both temporal operators and Boolean connectives (using Post's lattice of sets of Boolean functions).

In this section, we classify temporal formulas according to their clausal normal form. We remind the reader that any $\mathcal{PTL}$-formula can be transformed to an equisatisfiable formula in SNF [Fisher 1991]. A formula in SNF is a conjunction of *initial clauses* (that define "initial conditions" at moment 0), *step clauses* (that define "transitions" between consecutive states), and *eventuality clauses* (ensuring that certain states are eventually reached). More precisely, for the time flow $\mathbb{Z}$, a formula in SNF is a conjunction of clauses of the form

$$\begin{array}{l} L_1 \vee \cdots \vee L_k, \\ \boxast \big((L_1 \wedge \cdots \wedge L_k) \rightarrow \bigcirc(L'_1 \vee \cdots \vee L'_m)\big), \\ \boxast \big((L_1 \wedge \cdots \wedge L_k) \rightarrow \Diamond_F L\big), \\ \boxast \big((L_1 \wedge \cdots \wedge L_k) \rightarrow \Diamond_P L\big), \end{array}$$

where $L, L_1, \dots, L_k, L'_1, \dots, L'_m$ are *literals*—that is, propositional variables or their negations—and $\bigcirc$ is shorthand for $\bigcirc_F$ (we will use this abbreviation throughout this section). By definition, we assume the empty disjunction to be $\bot$ and the empty conjunction to be $\top$. For example, the second clause with $m = 0$ reads $\boxast\,(L_1 \wedge \cdots \wedge L_k \rightarrow \bot)$.

The transformation to SNF is achieved by fixed-point unfolding and renaming [Fisher et al. 2001; Plaisted 1986]. Recall that an occurrence of a subformula is said to be *positive* if it is in the scope of an even number of negations. Now, as $p\,\mathcal{U}\,q$ is equivalent to $\bigcirc q \vee (\bigcirc p \wedge \bigcirc(p\,\mathcal{U}\,q))$, every positive occurrence of $p\,\mathcal{U}\,q$ in a given formula φ can be replaced by a fresh propositional variable r, with the following three clauses added as conjuncts to φ:

$$\boxast\,(r \rightarrow \bigcirc(q \vee p)), \qquad \boxast(r \rightarrow \bigcirc(q \vee r)) \quad \text{and} \quad \boxast(r \rightarrow \Diamond_F q).$$

The result is equisatisfiable with φ but does not contain positive occurrences of $p\,\mathcal{U}\,q$. Similarly, we can get rid of other temporal operators and transform the formula to SNF [Fisher et al. 2001].

We now define four types of fragments of $\mathcal{PTL}$, which are called $\mathcal{PTL}_{core}(\mathcal{X})$, $\mathcal{PTL}_{krom}(\mathcal{X})$, $\mathcal{PTL}_{horn}(\mathcal{X})$, and $\mathcal{PTL}_{bool}(\mathcal{X})$, where $\mathcal{X}$ has one of the following four forms: $\boxast$, $\bigcirc_F/\bigcirc_P$, $\Box_F/\Box_P$, or $\boxast$, $\bigcirc_F/\bigcirc_P$, or $\boxast$, $\Box_F/\Box_P$, or $\boxast$. $\mathcal{PTL}_{core}(\mathcal{X})$-formulas, φ, are constructed using the following grammar:

$$\begin{array}{lcl} \varphi & ::= & \psi \quad | \quad \boxast\,\psi \quad | \quad \varphi_1 \wedge \varphi_2, \\ \psi & ::= & \lambda_1 \rightarrow \lambda_2 \quad | \quad \lambda_1 \wedge \lambda_2 \rightarrow \bot, \\ \lambda & ::= & \bot \quad | \quad p \quad | \quad \bigstar\lambda, \quad \text{where } \bigstar \text{ is one of the operators in } \mathcal{X}. \end{array} \tag{core}$$

Definitions of the remaining three fragments differ only in the shape of ψ. In $\mathcal{PTL}_{krom}(\mathcal{X})$-formulas, ψ is a binary clause:

$$\psi \quad ::= \quad \lambda_1 \rightarrow \lambda_2 \quad | \quad \lambda_1 \wedge \lambda_2 \rightarrow \bot \quad | \quad \lambda_1 \vee \lambda_2. \tag{krom}$$

In $\mathcal{PTL}_{horn}(\mathcal{X})$-formulas, ψ is a Horn clause:

$$\psi \quad ::= \quad \lambda_1 \wedge \cdots \wedge \lambda_n \rightarrow \lambda, \tag{horn}$$

whereas in $\mathcal{PTL}_{bool}(\mathcal{X})$-formulas, ψ is an arbitrary clause:

$$\psi \quad ::= \quad \lambda_1 \wedge \cdots \wedge \lambda_n \rightarrow \lambda'_1 \vee \cdots \vee \lambda'_k. \tag{bool}$$

Note that if $\mathcal{X}$ contains $\Box$-operators, then the corresponding $\Diamond$-operators can be defined in the fragments $\mathcal{PTL}_{krom}(\mathcal{X})$ and $\mathcal{PTL}_{bool}(\mathcal{X})$.

Table III. Complexity of Clausal Fragments of $\mathcal{PTL}$

	$\boxast$, $\Box_F/\Box_P$, $\bigcirc_F/\bigcirc_P$	$\boxast$, $\bigcirc_F/\bigcirc_P$	$\boxast$, $\Box_F/\Box_P$	$\boxast$
Bool	PSpace	PSpace	NP	NP
Horn	PSpace	PSpace	PTime [$\leq$ Th. 5.5]	PTime
Krom	NP [$\leq$ Th. 5.1]	NP	NP [$\geq$ Th. 5.8]	NLogSpace
core	NP	NP [$\geq$ Th. 5.4]	PTime [$\geq$Th. 5.7]	NLogSpace

Table III shows how the complexity of the satisfiability problem for $\mathcal{PTL}$-formulas depends on the type of the underlying propositional clauses and the available temporal operators. The PSpace upper bound is well known; the matching lower bound can be obtained by a standard encoding of deterministic Turing machines with polynomial tape (cf. Theorem 4.4). The NP upper bound for $\mathcal{PTL}_{bool}(\boxast, \Box_F/\Box_P)$ follows from [Ono and Nakamura 1980]. The NLogSpace lower bound is trivial, and the matching upper bound follows from the complexity of the Krom formulas with the quantifier prefix of the form $\exists^*\forall$ [Börger et al. 1997] (a similar argument is used in Theorem 4.9). In the remainder of this section, we prove all other results in Table III. It is worth noting how the addition of $\bigcirc$ or $\neg$ increases the complexity of $\mathcal{PTL}_{horn}(\boxast, \Box_F/\Box_P)$ and $\mathcal{PTL}_{core}(\boxast, \Box_F/\Box_P)$.

Theorem 5.1. *The satisfiability problem for $\mathcal{PTL}_{krom}(\boxast, \bigcirc_F/\bigcirc_P, \Box_F/\Box_P)$-formulas is in* NP.

Proof. We proceed as follows. First, in Lemma 5.2, we give a satisfiability criterion for $\mathcal{PTL}$-formulas in terms of types—sets of propositions that occur in the given formula—and distances between them in temporal models. The number of types required is polynomial in the size of the given formula; the distances, however, are exponential, and although they can be represented in binary (in polynomial space), in general there is no polynomial algorithm that checks whether two adjacent types can be placed at a given distance (unless PTime = PSpace). In the remainder of the proof, we show that for formulas with binary clauses, this condition can be verified by constructing a polynomial number of polynomial arithmetic progressions (using unary automata). This results in a nondeterministic polynomial-time algorithm: guess types and distances between them, and then verify (in polynomial time) whether the types can be placed at the required distances.

Let φ' be a $\mathcal{PTL}_{krom}(\boxast, \bigcirc_F/\bigcirc_P, \Box_F/\Box_P)$-formula. By introducing fresh propositional variables if required, we can transform φ' (in polynomial time) to a formula

$$\varphi \quad = \quad \Psi \quad \wedge \quad \boxast\, \Phi, \tag{14}$$

where Ψ contains no temporal operators and Φ contains no nested occurrences of temporal operators. Indeed, if φ' contains a conjunct ψ with a temporal λ, then we take a fresh propositional variable $\overline{\lambda}$, replace λ in ψ with $\overline{\lambda}$, and add to φ' a new conjunct $\boxast\,(\overline{\lambda} \leftrightarrow \lambda)$. In a similar way, we get rid of nested occurrences of temporal operators in Φ.

We will not distinguish between a set of formulas and the conjunction of its elements, and write $\boxast\,\Phi$ for $\bigwedge_{\chi \in \Phi} \boxast\,\chi$. As $\boxast\,(\lambda \vee \bigcirc_P\lambda')$ is equivalent to $\boxast\,(\bigcirc_F\lambda \vee \lambda')$, we can assume that Φ does not contain $\bigcirc_P$ (remember that we agreed to denote $\bigcirc_F$ by $\bigcirc$). We regard $\boxast$ inside Φ as defined by $\boxast\,\lambda = \Box_F\Box_P\lambda$. Thus, we assume that Φ contains only $\bigcirc$, $\Box_P$, and $\Box_F$ (which are not nested).

We first characterise the structure of models for formulas of the form (14) (with Ψ and Φ satisfying those conditions). It should be noted that this structure only depends on φ being of that form (cf. Gabbay et al. [1994, 2003] and references therein) and does not depend on whether or not Ψ and Φ are sets of binary clauses. To this end, for each $\Box_F L$ in φ, we take a fresh propositional variable, denoted $\overline{\Box_F L}$ and called the *surrogate* of

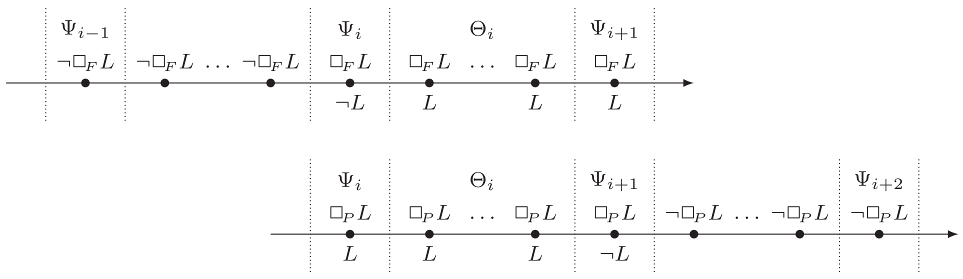

Fig. 3. Conditions **(B₂)**, **(B₄)**, and **(B₆)** in Lemma 5.2.

$\Box_F L$; likewise, for each $\Box_P L$, we take its surrogate $\overline{\Box_P L}$. Let $\overline{\Phi}$ be the result of replacing $\Box$-subformulas in Φ by their surrogates. It should be clear that φ is equisatisfiable with

$$\overline{\varphi} \quad = \quad \Psi \wedge \boxast \overline{\Phi} \quad \wedge \quad \bigwedge_{\Box_F L \text{ occurs in } \Phi} \boxast (\overline{\Box_F L} \leftrightarrow \Box_F L) \quad \wedge \quad \bigwedge_{\Box_P L \text{ occurs in } \Phi} \boxast (\overline{\Box_P L} \leftrightarrow \Box_P L).$$

By a *type* for $\overline{\varphi}$, we mean any set of literals that contains either p or $\neg p$, for each variable p in $\overline{\varphi}$ (including the surrogates $\overline{\Box_F L}$ and $\overline{\Box_P L}$).

LEMMA 5.2. *The formula $\overline{\varphi}$ is satisfiable iff there exist $k+5$ integers*

$$m_0 < m_1 < \cdots < m_{k+4}$$

(where k does not exceed the number of $\Box_F L$ and $\Box_P L$) and a sequence $\Psi_0, \Psi_1, \ldots, \Psi_{k+4}$ of types for $\overline{\varphi}$ satisfying the following conditions (see Figure 3):

(B₀) *$m_{i+1} - m_i < 2^{|\overline{\varphi}|}$, for $0 \leq i < k+4$;*
(B₁) *there exists ℓ_0 such that $0 \leq \ell_0 \leq k+4$ and $\Psi \wedge \Psi_{\ell_0}$ is consistent;*
(B₂) *for each i, $0 \leq i < k+4$, and each $\Box_F L$ in Φ,*

$$\textit{if } \overline{\Box_F L} \in \Psi_i \textit{ then } L, \overline{\Box_F L} \in \Psi_{i+1}, \qquad \textit{and} \qquad \textit{if } \overline{\Box_F L} \in \Psi_{i+1} \setminus \Psi_i \textit{ then } L \notin \Psi_{i+1};$$

(B₃) *there exists $\ell_F < k+4$ such that $\Psi_{\ell_F} = \Psi_{k+4}$ and, for each $\Box_F L$ in Φ,*

$$\textit{if } \overline{\Box_F L} \notin \Psi_{\ell_F} \textit{ then } L \notin \Psi_j, \textit{ for some } j \geq \ell_F;$$

(B₄) *for each i, $0 < i \leq k+4$, and each $\Box_P L$ in Φ,*

$$\textit{if } \overline{\Box_P L} \in \Psi_i \textit{ then } L, \overline{\Box_P L} \in \Psi_{i-1}, \qquad \textit{and} \qquad \textit{if } \overline{\Box_P L} \in \Psi_{i-1} \setminus \Psi_i \textit{ then } L \notin \Psi_{i-1};$$

(B₅) *there exists $\ell_P > 0$ such that $\Psi_{\ell_P} = \Psi_0$ and, for each $\Box_P L$ in Φ,*

$$\textit{if } \overline{\Box_P L} \notin \Psi_{\ell_P} \textit{ then } L \notin \Psi_j, \textit{ for some } j \leq \ell_P;$$

(B₆) *for all i, $0 \leq i < k+4$, the following formula is consistent:*

$$\Psi_i \quad \wedge \quad \bigwedge_{j=1}^{m_{i+1}-m_i-1} \bigcirc^j \Theta_i \quad \wedge \quad \bigcirc^{m_{i+1}-m_i} \Psi_{i+1} \quad \wedge \quad \boxast \overline{\Phi}, \tag{15}$$

where $\bigcirc^j \Psi$ is the result of attaching j operators $\bigcirc$ to each literal in Ψ and

$$\Theta_i \quad = \quad \{L, \overline{\Box_F L} \mid \overline{\Box_F L} \in \Psi_i\} \cup \{\neg\overline{\Box_F L} \mid \overline{\Box_F L} \notin \Psi_i\} \cup$$
$$\{L, \overline{\Box_P L} \mid \overline{\Box_P L} \in \Psi_{i+1}\} \cup \{\neg\overline{\Box_P L} \mid \overline{\Box_P L} \notin \Psi_{i+1}\}.$$

PROOF. ($\Rightarrow$) Let $\mathfrak{M}, 0 \models \overline{\varphi}$. Denote by $\Psi(m)$ the type for $\overline{\varphi}$ containing all literals that hold at m in $\mathfrak{M}$. As the number of types is finite, there is $m_F > 0$ such that each type in the sequence $\Psi(m_F), \Psi(m_F + 1), \dots$ appears infinitely often; similarly, there is $m_P < 0$ such that each type in the sequence $\Psi(m_P), \Psi(m_P - 1), \dots$ appears infinitely often. Then, for each subformula $\Box_F L$ of Φ, we have one of the three options: (1) L is always true in $\mathfrak{M}$, in which case we set $m_{\Box_F L} = 0$; (2) there is $m_{\Box_F L}$ such that $\mathfrak{M}, m_{\Box_F L} \models \neg L \wedge \Box_F L$, in which case $m_P < m_{\Box_F L} < m_F$; or (3) $\Box_F L$ is always false in $\mathfrak{M}$, in which case L is false infinitely often after m_F, and so there is $m_{\Box_F L} \geq m_F$ such that $\mathfrak{M}, m_{\Box_F L} \models \neg L$. Symmetrically, for each subformula $\Box_P L$ of Φ, (1) L is always true in $\mathfrak{M}$, in which case we set $m_{\Box_P L} = 0$; or (2) there is an $m_{\Box_P L}$ such that $m_P < m_{\Box_P L} < m_F$ and $\mathfrak{M}, m_{\Box_P L} \models \neg L \wedge \Box_P L$; or (3) $\Box_P L$ is always false in $\mathfrak{M}$, in which case there is $m_{\Box_P L} \leq m_P$ such that $\mathfrak{M}, m_{\Box_P L} \models \neg L$. Let $m_1 < m_2 < \cdots < m_{k+3}$ be an enumeration of the set

$$\{0, m_P, m_F\} \;\cup\; \{m_{\Box_F L} \mid \Box_F L \text{ occurs in } \Phi\} \;\cup\; \{m_{\Box_P L} \mid \Box_P L \text{ occurs in } \Phi\}.$$

Let $m_{k+4} > m_{k+3}$ be such that $\Psi(m_{k+4}) = \Psi(m_F)$ and let $m_0 < m_1$ be such that $\Psi(m_0) = \Psi(m_P)$. We then set $\Psi_i = \Psi(m_i)$, for $0 \leq i \leq k+4$. Let ℓ_0, ℓ_P, and ℓ_F be such that $m_{\ell_0} = 0$, $m_{\ell_P} = m_P$, and $m_{\ell_F} = m_F$. It should be clear that **($\mathbf{B_1}$)** through **($\mathbf{B_6}$)** hold. Finally, given a model of $\overline{\varphi}$ with two moments m and n such that the types at m and n coincide, we can construct a new model for $\overline{\varphi}$ by "removing" the states i with $m \leq i < n$. Since the number of distinct types is bounded by $2^{|\overline{\varphi}|}$, by repeated applications of this construction we can further ensure **($\mathbf{B_0}$)**.

($\Leftarrow$) We construct a model $\mathfrak{M}$ of $\overline{\varphi}$ by taking finite cuts of the models $\mathfrak{M}_i$ of the formulas in **($\mathbf{B_6}$)**: between the moments m_0 and m_{k+4}, the model $\mathfrak{M}$ coincides with the models $\mathfrak{M}_0, \dots, \mathfrak{M}_{k+3}$ so that at the moment m_i in $\mathfrak{M}$ we align the moment 0 of $\mathfrak{M}_i$, and at the moment m_{i+1} we align the moment $m_{i+1} - m_i$ of $\mathfrak{M}_i$, which coincides with the moment 0 of $\mathfrak{M}_{i+1}$ because both are defined by Ψ_{i+1}; before the moment m_0, the model $\mathfrak{M}$ repeats infinitely often its own fragment between m_0 and m_{ℓ_P}, and after m_{k+4} it repeats infinitely often its fragment between m_{ℓ_F} and m_{k+4} (both fragments contain more than one state). It is readily seen that $\mathfrak{M}, m_{\ell_0} \models \overline{\varphi}$. $\Box$

By Lemma 5.2, if we provide a polynomial-time algorithm for verifying **($\mathbf{B_6}$)**, we can check satisfiability of $\overline{\varphi}$ in NP. Indeed, it suffices to guess $k+5$ types for $\overline{\varphi}$ and $k+4$ natural numbers $n_i = m_{i+1} - m_i$, for $0 \leq i < k+4$, whose binary representation, by **($\mathbf{B_0}$)**, is polynomial in $|\overline{\varphi}|$. It is easy to see that **($\mathbf{B_1}$)** through **($\mathbf{B_5}$)** can be checked in polynomial time. We show now that **($\mathbf{B_6}$)** can also be verified in polynomial time for $\mathcal{PTL}_{krom}(\circledast, \bigcirc_F/\bigcirc_P, \Box_F/\Box_P)$-formulas.

Our problem is as follows: given a number $n \geq 0$ (in binary), types Ψ and Ψ', a set Θ of literals, and a set Φ of binary clauses of the form $D_1 \vee D_2$, where the D_i are temporal literals p, $\neg p$, $\bigcirc p$, or $\neg \bigcirc p$, decide whether there is a model satisfying

$$\Psi \quad \wedge \quad \bigwedge_{k=1}^{n-1} \bigcirc^k \Theta \quad \wedge \quad \bigcirc^n \Psi' \quad \wedge \circledast\, \Phi. \tag{16}$$

In what follows, we write $\psi_1 \models \psi_2$ as a shorthand for "in every $\mathfrak{M}$, if $\mathfrak{M}, 0 \models \psi_1$ then $\mathfrak{M}, 0 \models \psi_2$." For $0 \leq k \leq n$, we set:

$$F^k_\Phi(\Psi) = \{L' \mid L \wedge \circledast\, \Phi \models \bigcirc^k L', \text{ for } L \in \Psi\},$$
$$P^k_\Phi(\Psi') = \{L \mid \bigcirc^k L' \wedge \circledast\, \Phi \models L, \text{ for } L' \in \Psi'\}.$$

LEMMA 5.3. *Formula* (16) *is satisfiable iff the following conditions hold:*

($\mathbf{L_1}$) $F^0_\Phi(\Psi) \subseteq \Psi$, $F^n_\Phi(\Psi) \subseteq \Psi'$ *and* $P^0_\Phi(\Psi') \subseteq \Psi'$, $P^n_\Phi(\Psi') \subseteq \Psi$;

($\mathbf{L_2}$) $\neg L \notin F^k_\Phi(\Psi)$ *and* $\neg L \notin P^{n-k}_\Phi(\Psi')$, *for all* $L \in \Theta$ *and* $0 < k < n$.

PROOF. It should be clear that if (16) is satisfiable, then the preceding conditions hold. For the converse direction, observe that if $L' \in F^k_\Phi(\Psi)$, then since Φ is a set of binary clauses, there is a sequence of $\bigcirc$-prefixed literals $\bigcirc^{k_0} L_0 \leadsto \bigcirc^{k_1} L_1 \leadsto \cdots \leadsto \bigcirc^{k_m} L_m$ such that $k_0 = 0$, $L_0 \in \Psi$, $k_m = k$, $L_m = L'$, each k_i is between 0 and n, and the $\leadsto$ relation is defined by taking $\bigcirc^{k_i} L_i \leadsto \bigcirc^{k_{i+1}} L_{i+1}$ just in one of the three cases: $k_{i+1} = k_i$ and $L_i \to L_{i+1} \in \Phi$ or $k_{i+1} = k_i + 1$ and $L_i \to \bigcirc L_{i+1} \in \Phi$ or $k_{i+1} = k_i - 1$ and $\bigcirc L_i \to L_{i+1} \in \Phi$ (we assume, for example, that $\neg q \to \neg p \in \Phi$ whenever Φ contains $p \to q$). So, suppose that conditions **(L$_1$)** through **(L$_2$)** hold. We construct an interpretation satisfying (16). By **(L$_1$)**, both $\Psi \wedge \boxast\Phi$ and $\bigcirc^n\Psi' \wedge \boxast\Phi$ are consistent. So, let $\mathfrak{M}_\Psi$ and $\mathfrak{M}_{\Psi'}$ be such that $\mathfrak{M}_\Psi, 0 \models \Psi \wedge \boxast\,\Psi$ and $\mathfrak{M}_\Psi, n \models \Psi' \wedge \boxast\,\Psi$, respectively. Let $\mathfrak{M}$ be an interpretation that coincides with $\mathfrak{M}_\Psi$ for all moments $k \leq 0$ and with $\mathfrak{M}_{\Psi'}$ for all $k \geq n$; for the remaining k, $0 < k < n$, it is defined as follows. First, for each $p \in \Theta$, we make p true at k, and, for each $\neg p \in \Theta$, we make p false at k; such an assignment exists due to **(L$_2$)**. Second, we extend the assignment by making L true at k if $L \in F^k_\Phi(\Psi) \cup P^{n-k}_\Phi(\Psi')$. Observe that we have $\{p, \neg p\} \not\subseteq F^k_\Phi(\Psi) \cup P^{n-k}_\Phi(\Psi')$: for otherwise $L \wedge \boxast\Phi \models \bigcirc^k p$ and $\bigcirc^{n-k} L' \wedge \boxast\Phi \models \neg p$, for some $L \in \Psi$ and $L' \in \Psi'$, whence $L \wedge \boxast\Phi \models \bigcirc^n \neg L'$, contrary to **(L$_1$)**. In addition, by **(L$_2$)**, any assignment extension at this stage does not contradict the choices made due to Θ. Finally, all propositional variables not covered in the previous two cases get their values from $\mathfrak{M}_\Psi$ (or $\mathfrak{M}_{\Psi'}$). We note that the last choice does not depend on the assignment that is fixed by taking account of the consequences of $\boxast\Phi$ with Ψ, Ψ' and Θ (because if the value of a variable depended on those sets of literals, the respective literal would be among the logical consequences and would have been fixed before). $\square$

Thus, it suffices to show that conditions **(L$_1$)** and **(L$_2$)** can be checked in polynomial time. First, we claim that there is a polynomial-time algorithm that, given a set Φ of binary clauses of the form $D_1 \vee D_2$, constructs a set Φ^* of binary clauses that is "sound and complete" in the following sense:

(S$_1$) $\boxast\,\Phi^* \models \boxast\,\Phi$;
(S$_2$) if $\boxast\,\Phi \models \boxast\,(L \to \bigcirc^k L_k)$ then either $k = 0$ and $L \to L_0 \in \Phi^*$, or $k \geq 1$ and there are $L_0, L_1, \dots, L_{k-1}$ with $L = L_0$ and $L_i \to \bigcirc L_{i+1} \in \Phi^*$, for $0 \leq i < k$.

Intuitively, the set Φ^* makes explicit the consequences of $\boxast\Phi$ and can be constructed in time $(2|\Phi|)^2$ (the number of temporal literals in Φ^* is bounded by the doubled length $|\Phi|$ of Φ as each of its literals can only be prefixed by $\bigcirc$). Indeed, we start from Φ and, at each step, add $D_1 \vee D_2$ to Φ if it contains both $D_1 \vee D$ and $\neg D \vee D_2$; we also add $L_1 \vee L_2$ if Φ contains $\bigcirc L_1 \vee \bigcirc L_2$ (and vice versa). This procedure is sound since we only add consequences of $\boxast\Phi$; completeness follows from the completeness proof for temporal resolution [Fisher et al. 2001, Section 6.3].

Our next step is to encode Φ^* by means of unary automata. For literals L and L', consider a nondeterministic finite automaton $\mathfrak{A}_{L,L'}$ over $\{0\}$, whose states are the literals of Φ^*, L is the initial and L' is the only accepting state, and the transition relation is $\{(L_1, L_2) \mid L_1 \to \bigcirc L_2 \in \Phi^*\}$. By **(S$_1$)** and **(S$_2$)**, for all $k > 0$, we have

$$\mathfrak{A}_{L,L'} \text{ accepts } 0^k \quad\text{iff}\quad \boxast\,\Phi \models \boxast\,(L \to \bigcirc^k L').$$

Then, both $F^k_\Phi(\Psi)$ and $P^k_\Phi(\Psi')$, for $k > 0$, can be defined in terms of the language of $\mathfrak{A}_{L,L'}$:

$$F^k_\Phi(\Psi) = \big\{L' \mid \mathfrak{A}_{L,L'} \text{ accepts } 0^k, \text{ for } L \in \Psi\big\},$$

$$P^k_\Phi(\Psi') = \big\{L \mid \mathfrak{A}_{\neg L,\neg L'} \text{ accepts } 0^k, \text{ for } L' \in \Psi'\big\}$$

(recall that $\bigcirc^k L' \to L$ is equivalent to $\neg L \to \bigcirc^k \neg L'$). Note that the numbers n and k in conditions **(L$_1$)** and **(L$_2$)** are in general exponential in the length of Φ, and therefore the automata $\mathfrak{A}_{L,L'}$ do not immediately provide a polynomial-time procedure for checking

	1	2	3	4	5	**6**	7	8	9	**10**	11	12	13	14	**15**	**16**	17	18	19	20	**21**	22	23	24	**25**	26	27	28	29	**30**
2	**1**	0	1	0	1	**0**	1	0	1	**0**	1	0	1	0	**1**	**0**	1	0	1	0	**1**	0	1	0	**1**	0	1	0	1	**0**
3	**1**		0	1		**0**	1		0	**1**		0	1		**0**	**1**		0	1		**0**	1		0	**1**		0	1		**0**
5	**1**				0	**1**				**0**	1				**0**	**1**				0	**1**				**0**	1				**0**

Fig. 4. Natural numbers encoding assignments for three variables: p_1, p_2, p_3 (shown in boldface).

these conditions: although it can be shown that if **(L₂)** does not hold then it fails for a polynomial number k, this is not the case for **(L₁)**, which requires the accepting state to be reached in a fixed (exponential) number of transitions. Instead, we use the Chrobak normal form [Chrobak 1986] to decompose the automata into a polynomial number of polynomial-sized arithmetic progressions (which can have an exponential common period; cf. the proof of Theorem 5.4).

It is known that every N-state unary automaton $\mathfrak{A}$ can be converted (in polynomial time) into an equivalent automaton in Chrobak normal form (e.g., by using Martinez's algorithm [To 2009]), which has $O(N^2)$ states and gives rise to M arithmetic progressions $a_1 + b_1\mathbb{N}, \dots, a_M + b_M\mathbb{N}$, where $a_i + b_i\mathbb{N} = \{a_i + b_i m \mid m \in \mathbb{N}\}$, such that

—$M \leq O(N^2)$ and $0 \leq a_i, b_i \leq N$, for $1 \leq i \leq M$;
—$\mathfrak{A}$ accepts 0^k iff $k \in a_i + b_i\mathbb{N}$, for some $1 \leq i \leq M$.

By construction, the number of arithmetic progressions is quadratic in the length of Φ.

We are now in a position to give a polynomial-time algorithm for checking **($\mathbf{L_1}$)** and **($\mathbf{L_2}$)**, which requires solving Diophantine equations. In **($\mathbf{L_2}$)**, for example, to verify that for each $p \in \Theta$, we have $\neg p \notin F^k_\Phi(\Psi)$, for all $0 < k < n$, we take the automata $\mathfrak{A}_{L,\neg p}$, for $L \in \Psi$ and transform them into the Chrobak normal form to obtain arithmetic progressions $a_i + b_i\mathbb{N}$, for $1 \leq i \leq M$. Then, there is k, $0 < k < n$, with $\neg p \in F^k_\Phi(\Psi)$ iff one of the equations $a_i + b_i m = k$ has an integer solution with $0 < k < n$. The latter can be verified by taking the integer $m = \lfloor -a_i/b_i \rfloor$ and checking whether either $a_i + b_i m$ or $a_i + b_i(m+1)$ belongs to the open interval $(0, n)$, which can clearly be done in polynomial time. This completes the proof of Theorem 5.1.

We can establish the matching lower bound for $\mathcal{PTL}_{core}(\circledast, \bigcirc_F/\bigcirc_P)$-formulas by using a result on the complexity of deciding inequality of regular languages over singleton alphabets [Stockmeyer and Meyer 1973]. In the following theorem, we give a more direct reduction of the NP-complete problem 3SAT and repeat the argument of Stockmeyer and Meyer [1973, Theorem 6.1] to construct a small number of arithmetic progressions (each with a small initial term and common difference) that give rise to models of exponential size:

THEOREM 5.4. *The satisfiability problem for $\mathcal{PTL}_{core}(\circledast, \bigcirc_F/\bigcirc_P)$-formulas is* NP-*hard.*

PROOF. The proof is by reduction of 3SAT [Papadimitriou 1994]. Let $f = \bigwedge_{i=1}^n C_i$ be a 3CNF with m variables $p_1, \dots, p_m$ and n clauses $C_1, \dots, C_n$. By a propositional assignment for f, we understand a function $\sigma : \{p_1, \dots, p_m\} \to \{0, 1\}$. We will represent such assignments by sets of positive natural numbers. More precisely, let $P_1, \dots, P_m$ be the first m prime numbers; it is known that P_m does not exceed $O(m^2)$ [Apostol 1976]. We say that a natural number k represents an assignment σ if k is equivalent to $\sigma(p_i)$ modulo P_i, for all i, $1 \leq i \leq m$. Not every natural number represents an assignment. Consider the following arithmetic progressions:

$$j + P_i \cdot \mathbb{N}, \qquad \text{for } 1 \leq i \leq m \text{ and } 2 \leq j < P_i. \tag{17}$$

Every element of $j + P_i \cdot \mathbb{N}$ is equivalent to j modulo P_i, and so, since $j \geq 2$, cannot represent an assignment. Moreover, every natural number that cannot represent an assignment belongs to one of these arithmetic progressions (see Figure 4).

Let C_i be a clause in f—for example, $C_i = p_{i_1} \vee \neg p_{i_2} \vee p_{i_3}$. Consider the following progression:

$$P^1_{i_1} P^0_{i_2} P^1_{i_3} + P_{i_1} P_{i_2} P_{i_3} \cdot \mathbb{N}. \tag{18}$$

Then, a natural number represents an assignment making C_i true iff it *does not* belong to progressions (17) and (18). Thus, a natural number represents a satisfying assignment for f iff does not belong to any of the progressions of the form (17) and (18), for clauses in f.

To complete the proof, we show that the defined arithmetic progressions can be encoded in $\mathcal{PTL}_{core}(\circledast, \bigcirc_F/\bigcirc_P)$. We take a propositional variable d, which will be shared among many formulas that follow. Given an arithmetic progression $a + b\mathbb{N}$ (with $a \geq 0$ and $b > 0$), consider the formula

$$\theta_{a,b} = u_0 \wedge \bigwedge_{j=1}^{a} \circledast(u_{j-1} \to \bigcirc u_j) \wedge \circledast(u_a \to v_0) \wedge \bigwedge_{j=1}^{b} \circledast(v_{j-1} \to \bigcirc v_j) \wedge \circledast(v_b \to v_0) \wedge \circledast(v_0 \to d),$$

where $u_0, \dots, u_a$ and $v_0, \dots, v_b$ are fresh propositional variables. It is not hard to see that in every model satisfying $\theta_{a,b}$ at moment 0, d is true at moment $k \geq 0$ whenever k belongs to $a + b\mathbb{N}$.

So, we take $\theta_{a,b}$ for each of the arithmetic progressions (17) and (18) and add two formulas, $p \wedge \circledast(\bigcirc p \to p) \wedge \circledast(p \to d)$ and $\circledast d \to \bot$, which ensure that d and a fresh variable p are true at all $k \leq 0$ but d is not true at all moments. The size of the resulting conjunction of $\mathcal{PTL}_{core}(\circledast, \bigcirc_F/\bigcirc_P)$-formulas is $O(n \cdot m^6)$. One can check that it is satisfiable iff f is satisfiable. $\square$

THEOREM 5.5. *The satisfiability problem for $\mathcal{PTL}_{horn}(\circledast, \Box_F/\Box_P)$-formulas is in* PTIME.

PROOF. Without loss of generality, we can assume that $\circledast$ does not occur in the formulas of the form λ and that $\Box_F$, $\Box_P$ are applied only to variables. Now, observe that every satisfiable $\mathcal{PTL}_{horn}(\circledast, \Box_F/\Box_P)$-formula φ is satisfied in a model with a short prefix (of length linear in $|\varphi|$) followed by a loop of length 1 (cf. Lemma 5.2). More precisely, let $\mathfrak{M}, 0 \models \varphi$. Similarly to the proof of Lemma 5.2, for each subformula $\Box_F p_i$ of φ, we have only three possible choices: if $\Box_F p_i$ is always true or always false, we set $m_{\Box_F p_i} = 0$; otherwise, there is $m_{\Box_F p_i}$ with $\mathfrak{M}, m_{\Box_F p_i} \models \neg p_i \wedge \Box_F p_i$. Symmetrically, we take all moments $m_{\Box_P p_i}$ for all $\Box_P p_i$ in φ. Consider the following set

$$\{0\} \quad \cup \quad \{m_{\Box_P p_i} \mid \Box_P p_i \text{ occurs in } \varphi\} \quad \cup \quad \{m_{\Box_F p_i} \mid \Box_F p_i \text{ occurs in } \varphi\}$$

and suppose that it consists of the numbers $m_{-l} < \cdots < m_{-1} < m_0 < m_1 < \cdots < m_k$ with $m_0 = 0$. Let N be the number of $\Box_P p_i$ and $\Box_F p_i$ occurring in φ plus 1. We extend the sequence by taking $m_i = m_k + 1$, for $k < i \leq N$, and $m_{-i} = m_{-l} - 1$, for $l < i \leq N$. Therefore, $\mathfrak{M}, m_N \models \Box_F p_i$ iff $\mathfrak{M}, m_N \models p_i$ (and symmetrically for $\Box_P p_i$ at m_{-N}). Let $\mathfrak{M}'$ be defined as follows:

$$\mathfrak{M}', n \models p_i \quad \text{iff} \quad \begin{cases} \mathfrak{M}, m_{-N} \models p_i, & \text{if } n < -N, \\ \mathfrak{M}, m_n \models p_i, & \text{if } -N \leq n \leq N, \\ \mathfrak{M}, m_N \models p_i, & \text{if } n > N. \end{cases}$$

It can be seen that $\mathfrak{M}', 0 \models \varphi$.

It remains to encode the existence of such a model by means of propositional Horn formulas, as Horn-SAT is known to be PTIME-complete. To this end, for each propositional variable p_i, we take $2N+1$ variables p_i^m, for $-N \leq m \leq N$. In addition, for each formula $\Box_F p_i$, we take $2N + 1$ propositional variables, denoted $(\Box_F p_i)^m$, for $-N \leq m \leq N$, and similarly, for each $\Box_P p_i$, we take variables $(\Box_P p_i)^m$. Then, each clause $\lambda_1 \wedge \cdots \wedge \lambda_n \to \lambda$ in φ gives rise to the propositional clause

$$\lambda_1^0 \wedge \cdots \wedge \lambda_n^0 \to \lambda^0$$

and each $\boxast(\lambda_1 \wedge \cdots \wedge \lambda_n \to \lambda)$ in φ gives rise to $2N+1$ clauses

$$\lambda_1^m \wedge \cdots \wedge \lambda_n^m \to \lambda^m, \qquad \text{for } -N \leq m \leq N.$$

Additionally, we need clauses that describe the semantics of $\Box_F p_i$ in $\mathfrak{M}'$ at m, for $-N \leq m < N$:

$$(\Box_F p_i)^m \to (\Box_F p_i)^{m+1}, \qquad (\Box_F p_i)^m \to p_i^{m+1}, \qquad (\Box_F p_i)^{m+1} \wedge p_i^{m+1} \to (\Box_F p_i)^m,$$

and clauses that describe the semantics of $\Box_F p_i$ in $\mathfrak{M}'$ at moment N:

$$(\Box_F p_i)^N \to p_i^N, \qquad p_i^N \to (\Box_F p_i)^N,$$

and symmetric clauses for each $\Box_P p_i$ in φ. It is not hard to show that every satisfying assignment for the set of the preceding clauses gives rise to a model $\mathfrak{M}'$ of φ and, conversely, every model $\mathfrak{M}'$ of φ with the structure as described previously gives rise to a satisfying assignment for this set of clauses. □

Remark 5.6. In order to obtain Theorem 4.8, one can extend the preceding proof to formulas of the form $\varphi' \wedge \varphi''$, where φ' is a $\mathcal{PTL}_{horn}(\boxast, \Box_F/\Box_P)$-formula and φ'' a conjunction of $\bigcirc^n p$, for propositional variables p. To this end, in the definition of the set M, one has to take 0 together with all n for which $\bigcirc^n p$ occurs in φ''; the number N is then equal to the number of those moments n plus the number of all $\Box_F p$ and $\Box_P p$ occurring in φ'. The rest of the construction remains the same.

THEOREM 5.7. *The satisfiability problem for $\mathcal{PTL}_{core}(\boxast, \Box_F/\Box_P)$-formulas is* PTIME-*hard.*

PROOF. The proof is by reduction of satisfiability of propositional Horn formulas with at most ternary clauses, which is known to be PTIME-complete [Papadimitriou 1994]. Let $f = \bigwedge_{i=1}^n C_i$ be such a formula. We define φ_f to be the conjunction of the following formulas:

p, for all clauses C_i of the form p,

$p \to \bot$, for all clauses C_i of the form $\neg p$,

$p \to q$, for all clauses C_i of the form $p \to q$,

$c_i \wedge (p \to \Box_F c_i) \wedge (q \to \Box_P c_i) \wedge (\boxast\, c_i \to r)$, for all clauses C_i of the form $p \wedge q \to r$,

where c_i is a fresh variable for each C_i. It is easy to see that f is satisfiable iff φ_f is satisfiable. □

THEOREM 5.8. *The satisfiability problem for $\mathcal{PTL}_{krom}(\boxast, \Box_F/\Box_P)$-formulas is* NP-*hard.*

PROOF. We proceed by reduction of the three-colourability problem. Given a graph $G = (V, E)$, we use variables $p_0, \ldots, p_4$ and v_i, for $v_i \in V$, to define the following $\mathcal{PTL}_{krom}(\boxast, \Box_F/\Box_P)$-formula:

$$\begin{aligned}\varphi_G \quad = \quad & p_0 \quad \wedge \bigwedge_{0 \leq i \leq 3} \boxast\,(p_i \to \Box_F p_{i+1}) \wedge \\ & \bigwedge_{v_i \in V} \boxast\,(p_0 \wedge \Box_F \neg v_i \to \bot) \quad \wedge \bigwedge_{v_i \in V} \boxast\,(p_4 \wedge v_i \to \bot) \quad \wedge \bigwedge_{(v_i, v_j) \in E} \boxast\,(v_i \wedge v_j \to \bot).\end{aligned}$$

Intuitively, the first four conjuncts choose, for each vertex v_i of the graph, a moment $1 \leq n_i \leq 3$; the last one makes sure that $n_i \neq n_j$ in case v_i and v_j are connected by an edge in G. We show that φ_G is satisfiable iff G is three colourable. Suppose that $c : V \to \{1, 2, 3\}$ is a colouring function for G. Define $\mathfrak{M}$ by setting $\mathfrak{M}, n \models v_i$ just in case $c(v_i) = n$, for $v_i \in V$, and $\mathfrak{M}, n \models p_i$ iff $n \geq i$, for $0 \leq i \leq 4$. Clearly, $\mathfrak{M}, 0 \models \varphi_G$. Conversely, if $\mathfrak{M}, 0 \models \varphi_G$ then, for each $v_i \in V$, there is $n_i \in \{1, 2, 3\}$ with $\mathfrak{M}, n_i \models v_i$ and $\mathfrak{M}, n_i \not\models v_j$ whenever $(v_i, v_j) \in E$. Thus, $c : v_i \mapsto n_i$ is a colouring function. □

6. *DL-Lite* WITH TEMPORALISED ROLES

Now we investigate the complexity of extensions of $\mathit{DL\text{-}Lite}_{bool}$ with temporalised roles of the form

$$R \ ::= \ S \ \mid \ S^{-} \ \mid \ \diamond R \ \mid \ \boxplus R,$$

where, as before, S is a flexible or rigid role name. Recall that the interpretation of $\diamond R$ and $\boxplus R$ is defined by taking $(\diamond R)^{\mathcal{I}(n)} = \bigcup_{k\in\mathbb{Z}} R^{\mathcal{I}(k)}$ and $(\boxplus R)^{\mathcal{I}(n)} = \bigcap_{k\in\mathbb{Z}} R^{\mathcal{I}(k)}$.

6.1. Directed Temporal Operators and Functionality: Undecidability

Our first result is negative. It shows, in fact, that any extension of $\mathit{DL\text{-}Lite}_{bool}$ with temporalised roles, functionality constraints on roles, and either the next-time operator $\bigcirc_F$ or both $\Box_F$ and $\Box_P$ on concepts is undecidable.

THEOREM 6.1. *Satisfiability of* $T^{*}_{X}\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ *and* $T^{*}_{FP}\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ *KBs is undecidable.*

PROOF. The proof is by reduction of the $\mathbb{N}\times\mathbb{N}$-tiling problem (e.g., see Börger et al. [1997]): given a finite set $\mathfrak{T}$ of tile types $T = (\mathit{up}(T), \mathit{down}(T), \mathit{left}(T), \mathit{right}(T))$, decide whether $\mathfrak{T}$ can tile the $\mathbb{N}\times\mathbb{N}$-grid—that is, whether there is a map $\tau : \mathbb{N}\times\mathbb{N} \to \mathfrak{T}$ such that $\mathit{up}(\tau(i, j)) = \mathit{down}(\tau(i, j+1))$ and $\mathit{right}(\tau(i, j)) = \mathit{left}(\tau(i+1, j))$, for all $(i, j) \in \mathbb{N}\times\mathbb{N}$. We assume that the colours of tiles in $\mathfrak{T}$ are natural numbers from 1 to k, for a suitable $k > 1$.

Consider first $T^{*}_{X}\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ and, given $\mathfrak{T}$, construct a KB $\mathcal{K}_{\mathfrak{T}} = (\mathcal{T}_{\mathfrak{T}}, \mathcal{A})$ such that $\mathcal{K}_{\mathfrak{T}}$ is satisfiable iff $\mathfrak{T}$ tiles the $\mathbb{N}\times\mathbb{N}$-grid. The temporal dimension will provide us with the horizontal axis of the grid. The vertical axis will be constructed using domain elements. Let R be a role such that

$$\geq 2 \diamond R \sqsubseteq \bot \quad \text{and} \quad \geq 2 \diamond R^{-} \sqsubseteq \bot. \tag{19}$$

In other words, if xRy at some moment, then there is no other y' with xRy' at any moment of time (and similarly for R^{-}). We generate a sequence of domain elements: first, we ensure that the concept $\exists R \sqcap \bigcirc_F \exists R$ is nonempty, which can be done by taking $\mathcal{A} = \{A(a)\}$ and adding

$$A \sqsubseteq \exists R \sqcap \bigcirc_F \exists R \tag{20}$$

to the TBox $\mathcal{T}_{\mathfrak{T}}$, and second, we add the following concept inclusion to $\mathcal{T}_{\mathfrak{T}}$ to produce a sequence:

$$\exists R^{-} \sqcap \bigcirc_F R^{-} \sqsubseteq \exists R \sqcap \bigcirc_F \exists R. \tag{21}$$

(The reason for generating the R-arrows at two consecutive moments of time will become clear later.) It is to be noted that the produced sequence may in fact be either a finite loop or an infinite chain of distinct elements. Now, let T be a fresh concept name for each $T \in \mathfrak{T}$, and let the concepts representing the tile types be disjoint:

$$T \sqcap T' \sqsubseteq \bot, \qquad \text{for } T \neq T'. \tag{22}$$

Right after the double R-arrows we place the first column of tiles:

$$\exists R^{-} \sqcap \bigcirc_F R^{-} \ \sqsubseteq \ \bigsqcup_{T\in\mathfrak{T}} \bigcirc_F\bigcirc_F T. \tag{23}$$

The second column of tiles, whose colours match the colours of the first one, is placed $k+1$ moments later; the third column is located $k+1$ moments after the second one, and so forth (see Figure 5):

$$T \ \sqsubseteq \ \bigsqcup_{T'\in\mathfrak{T} \text{ with } \mathit{right}(T)=\mathit{left}(T')} \bigcirc_F^{k+1} T', \qquad \text{for each } T \in \mathfrak{T}. \tag{24}$$

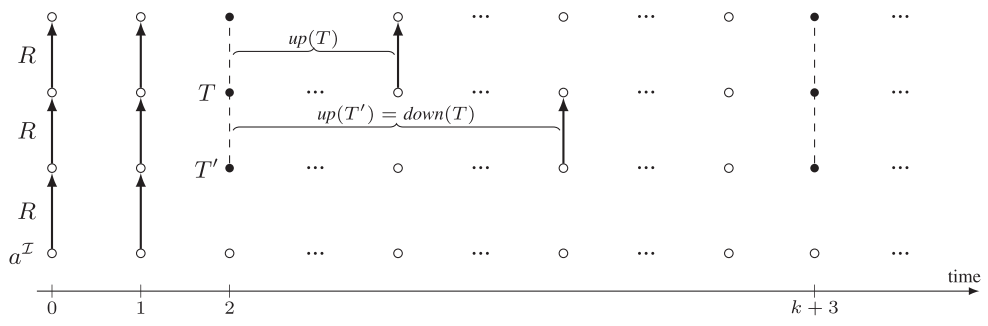

Fig. 5. Proof of Theorem 6.1: the structure of the $\mathbb{N} \times \mathbb{N}$ grid.

This gives an $\mathbb{N} \times \mathbb{N}$-grid of tiles with matching *left–right* colours. To ensure that the *up–down* colours in this grid also match, we use the double R-arrows at the beginning and place the columns of tiles $k+1$ moments apart from each other. Consider the following concept inclusions, for $T \in \mathfrak{T}$:

$$T \sqsubseteq \neg\exists R^-, \tag{25}$$

$$T \sqsubseteq \neg\bigcirc_F^i \exists R^-, \qquad \text{for } 1 \leq i \leq k \text{ with } i \neq down(T), \tag{26}$$

$$T \sqsubseteq \bigcirc_F^{up(T)} \exists R. \tag{27}$$

Inclusions (25), (22), and (26) ensure that between any two tiles $k+1$ moments apart, there may be only one incoming R-arrow. This means that after the initial double R-arrows, no other two consecutive R-arrows can occur. The exact position of the incoming R-arrow is uniquely determined by the *down*-colour of the tile, which together with (27) guarantees that this colour matches the *up*-colour of the tile below. Figure 5 illustrates the construction (the solid vertical arrows represent R).

Let $\mathcal{T}_{\mathfrak{T}}$ contain all of the concept inclusions defined earlier. It is not hard to check that $(\mathcal{T}_{\mathfrak{T}}, \mathcal{A})$ is satisfiable iff $\mathfrak{T}$ tiles the $\mathbb{N} \times \mathbb{N}$-grid.

The proof for T^*_{FP}*DL-Lite*$^{\mathcal{N}}_{bool}$ is much more involved. To encode the vertical axis of the $\mathbb{N} \times \mathbb{N}$-grid, we again use the role R satisfying the concept inclusions

$$\geq 2 \,\Diamond\!\!\!\!\diamond\, R \sqsubseteq \bot \quad \text{and} \quad \geq 2 \,\Diamond\!\!\!\!\diamond\, R^- \sqsubseteq \bot. \tag{28}$$

However, as $\bigcirc_F$ is not available in T^*_{FP}*DL-Lite*$^{\mathcal{N}}_{bool}$, we need a completely different construction to ensure that the tiles match in the horizontal dimension. Indeed, in the earlier proof (cf. (24)), we use $\bigcirc_F^n$ and disjunction to place a suitable tile to the right of any tile in the grid. Without the $\bigcirc_F$ operator, we use another role S (whose $\Diamond\!\!\!\!\diamond\, S$ is also inverse functional) and create special patterns to represent colours (as a natural number from 1 to k) similarly to the way we paired *up* and *down* colours earlier. To create patterns and refer to the "next moment," we use a trick similar to the one we used in the proof of Theorem 4.6: given a concept C and $n \geq 0$, let

$$\Diamond_F^{=n} C = \Diamond_F^n C \sqcap \neg\Diamond_F^{n+1} C \qquad \text{and} \qquad \Diamond_P^{=n} C = \Diamond_P^n C \sqcap \neg\Diamond_P^{n+1} C.$$

Note, however, that these $\Diamond_{F/P}^{=n} C$-operators can mark a domain element with C only once. So, every time we need a pattern, say of $\exists S$, of a certain length on a domain element, we create a new S-successor, use concepts bit_i (with various superscripts in the proof) to mark certain positions on that S-successor by means of the operators $\Diamond_{F/P}^{=i} bit_i$, and then "transfer" the markings back to our domain element via inclusions of the form $bit_i \sqsubseteq \exists S^-$ and $bit_i \sqsubseteq \neg\exists S^-$ with functional $\Diamond\!\!\!\!\diamond\, S^-$.

The rest of the proof is organised as follows. In Step 1, we create the structure of the horizontal axis on a fixed ABox element a. The structure consists of repeating blocks of

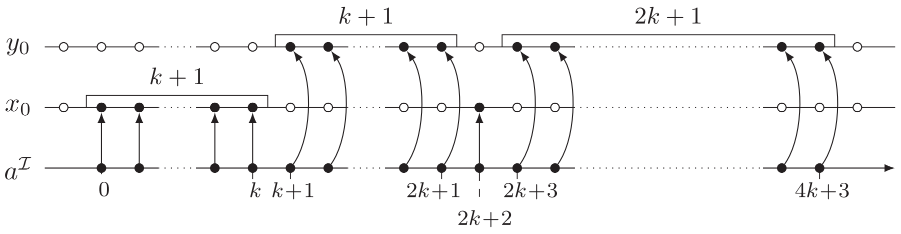

Fig. 6. The structure of the horizontal axis: x_0 is a V_1-successor of $a^{\mathcal{I}}$ and y_0 is a V_0-successor of $a^{\mathcal{I}}$.

length $4k+4$ (to represent the four colours of the tile); each block has a certain pattern of complementary V_0- and V_1-arrows (see Figure 8), which are arranged using the same technique as we outlined for S so that a has a "fan" of V_0-successors $(y_0, y_1, \dots)$ and a fan of V_1-successors $(x_0, x_1, \dots)$. Then, in Step 2, we create a sequence $z_0, z_1, \dots$ of R-successors to represent the vertical axis (see Figure 10) so that each of the z_i repeats the structure of the horizontal axis (shifted by $k+1$ with each new z_i) and places tiles on a fan of its own S-successors. The particular patterns of S-arrows within the repeating $4k+4$ blocks will then ensure that the *right–left* colours match (within the same fan) and, similarly, the patterns of R-arrows between the z_i will ensure that the *up–down* colours match.

Step 1. We encode the horizontal axis using the ABox $\mathcal{A} = \{A(a)\}$ and a number of concept inclusions with roles V_0, V_1, and concepts $bit_i^{V_1}$, for $1 \le i \le 2k+2$, and $bit_i^{V_0}$, for $1 \le i \le 3k+2$. Consider first the following concept inclusions:

$$A \sqsubseteq \exists V_1 \sqcap \Box_P \neg \exists V_1, \tag{29}$$

$$\geq 2 \circledast V_1^- \sqsubseteq \bot, \tag{30}$$

$$\exists V_1^- \sqcap \Box_P \neg \exists V_1^- \sqsubseteq \bigsqcap_{1 \le i \le 2k+2} \Diamond_F^{=i}\, bit_i^{V_1} \;\sqcap\; \Box_F^{2k+3} \neg \exists V_1^-, \tag{31}$$

$$bit_i^{V_1} \sqsubseteq \exists V_1^-, \qquad \text{for } 1 \le i \le k, \tag{32}$$

$$bit_{k+i}^{V_1} \sqsubseteq \neg \exists V_1^-, \qquad \text{for } 1 \le i \le k+1, \tag{33}$$

$$bit_{2k+2}^{V_1} \sqsubseteq \exists V_1^-. \tag{34}$$

Suppose that all of them hold in an interpretation $\mathcal{I}$. Then, by (29), $a^{\mathcal{I}}$ has a V_1-successor, say x_0, at moment 0 and no V_1-successor at any preceding moment. By (30), x_0 does not have a V_1-predecessor before 0, and so, by (31) through (34), x_0 has a V_1-predecessor at every moment i with $0 \le i \le k$ and $i = 2k+2$, and no V_1-predecessor at any other times. By (30), all of these V_1-predecessors must coincide with $a^{\mathcal{I}}$ (Figure 6). We also need similar concept inclusions for the role V_0:

$$A \sqsubseteq \Box_P \neg \exists V_0, \tag{35}$$

$$\geq 2 \circledast V_0^- \sqsubseteq \bot, \tag{36}$$

$$\exists V_0^- \sqcap \Box_P \neg \exists V_0^- \sqsubseteq \bigsqcap_{1 \le i \le 3k+2} \Diamond_F^{=i}\, bit_i^{V_0} \;\sqcap\; \Box_F^{3k+3} \neg \exists V_0^-, \tag{37}$$

$$bit_i^{V_0} \sqsubseteq \exists V_0^-, \qquad \text{for } 1 \le i \le k, \tag{38}$$

$$bit_{k+1}^{V_0} \sqsubseteq \neg \exists V_0^-, \tag{39}$$

$$bit_{k+1+i}^{V_0} \sqsubseteq \exists V_0^-, \qquad \text{for } 1 \le i \le 2k+1, \tag{40}$$

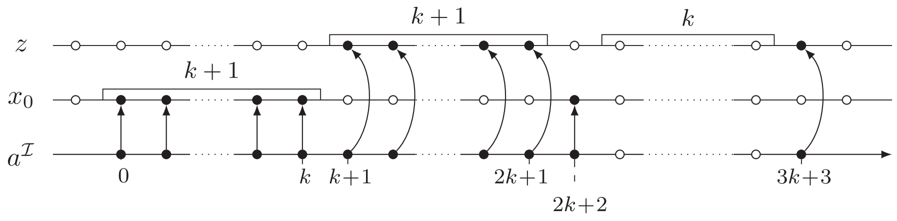

Fig. 7. A gap of k moments on the horizontal axis: both x_0 and z are V_1-successors of $a^{\mathcal{I}}$.

together with

$$A \sqsubseteq \Box_F(\exists V_1 \sqcup \exists V_0), \tag{41}$$

$$\exists V_0 \sqcap \exists V_1 \sqsubseteq \bot. \tag{42}$$

Suppose that all of them hold in $\mathcal{I}$. By (41), (42), at each moment after 0, $a^{\mathcal{I}}$ has either a V_0- or a V_1-successor. By (29), (42), and the preceding observations, $a^{\mathcal{I}}$ cannot have a V_0-successor in the interval between 0 and k. Suppose that y_0 is a V_0-successor of $a^{\mathcal{I}}$ at $k+1$ (that this is the case will be ensured by (43)). By (35), (36), y_0 has no V_0-predecessors before 0; so, by (37) through (40), y_0 has V_0-predecessors at the moments i with $k+1 \leq i \leq 2k+1$ and $2k+3 \leq i \leq 4k+3$ and no V_0-predecessors at other moments. By (36), all of these V_0-predecessors coincide with $a^{\mathcal{I}}$ (see Figure 6). We now show that if

$$\geq 2\, V_1 \sqsubseteq \bot \tag{43}$$

also holds in $\mathcal{I}$, then $a^{\mathcal{I}}$ has a V_0-successor at $k+1$. Indeed, suppose that $a^{\mathcal{I}}$ has a V_1-successor z at $k+1$. Then, by (29), the choice of x_0, and (43), z cannot be a V_1-successor of $a^{\mathcal{I}}$ at any moment before that. So, z must belong to the left-hand side concept of (31), which triggers the following pattern of V_1-successors of $a^{\mathcal{I}}$: x_0 at moments i with $0 \leq i \leq k$, z at i with $k+1 \leq i \leq 2k+1$, x_0 at $2k+2$ and z at $3k+3$ (Figure 7). This leaves only the moments i, for $2k+3 \leq i \leq 3k+2$, without any V_0- or V_1-successors. But in this case, $a^{\mathcal{I}}$ cannot have any V_0- or V_1-successor at $2k+3$. Indeed, such a V_0-successor z' would have no V_0-predecessor at any moment before $2k+3$, and so, by (36) through (40), would remain a V_0-successor of $a^{\mathcal{I}}$ for $k+1$ consecutive moments, which is impossible with only k available slots; by a similar argument and (43), $a^{\mathcal{I}}$ has no V_1-successor at $2k+3$.

Next, if in addition

$$\geq 2\, V_0 \sqsubseteq \bot \tag{44}$$

holds in $\mathcal{I}$, then $a^{\mathcal{I}}$ has a V_1-successor, x_1, at $4k+4$. Indeed, using (44) and an argument similar to the preceding one, one can show that if $a^{\mathcal{I}}$ has a V_0-successor z at $4k+4$, then z is different from y_1 and z cannot have V_0-predecessors before $4k+4$. But then the pattern of V_0-successors required by (37) through (40) would make it impossible for $a^{\mathcal{I}}$ to have any V_0- or V_1-successor at $6k+6$, where z has no V_0-predecessor.

Thus, we find ourselves in the same situation as at the very beginning of the construction but with x_1 in place of x_0. By repeating the same argument again and again, we obtain domain elements $x_0, x_1, \ldots$ and $y_0, y_1, \ldots$ of the interpretation $\mathcal{I}$, which are, respectively, V_1- and V_0-successors of $a^{\mathcal{I}}$ at the moments of time indicated in Figure 8 by black points and intervals.

Step 2. We are now in a position to encode the $\mathbb{N} \times \mathbb{N}$-tiling problem. Let us regard each $T \in \mathfrak{T}$ as a fresh concept name satisfying the disjointness concept inclusions

$$T \sqcap T' \sqsubseteq \bot, \qquad \text{for } T \neq T'. \tag{45}$$

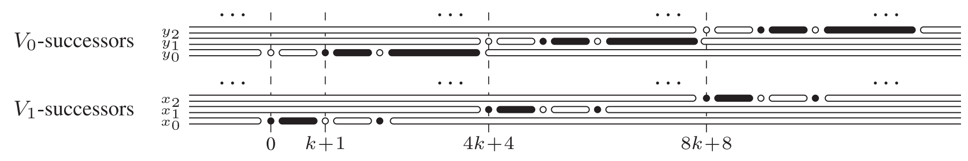

Fig. 8. V_0- and V_1-successors of a in a model of $\mathcal{K}_{\mathfrak{T}}$.

Consider the following concept inclusions:

$$A \sqsubseteq \exists R \sqcap \Box_P \neg \exists R, \tag{46}$$

$$\exists R^- \sqcap \Box_P \neg \exists R^- \sqsubseteq \Diamond_P^{=2k+1} \mathit{row\text{-}start}, \tag{47}$$

$$\mathit{row\text{-}start} \sqsubseteq \exists S \sqcap \Box_P \neg \exists S, \tag{48}$$

$$\exists S^- \sqcap \Box_P \neg \exists S^- \sqsubseteq \bigsqcup_{T \in \mathfrak{T}} T. \tag{49}$$

Intuitively, (46) says that a has an R-successor, say z_0, at the moment 0, and no R-successors before 0. Then, by (28), z_0 has no R-predecessors before 0. Axioms (47) through (49) make sure that z_0 has an S-successor, w, which is an instance of T at $-(2k+1)$, for some tile T. In this case, we say that T *is placed on* z_0 (rather than on w). Tiles will also be placed on domain elements having S-successors with a specific pattern of concepts $\exists S^-$ given by the following concept inclusions:

$$\geq 2 \diamondsuit S^- \sqsubseteq \bot, \tag{50}$$

$$\geq 2\, S \sqsubseteq \bot, \tag{51}$$

$$T \sqsubseteq \bigsqcap_{1 \leq i \leq 6k+4} \Diamond_F^{=i}\, \mathit{bit}_i^T \quad \sqcap \quad \Box_F^{6k+5} \neg \exists S^-, \tag{52}$$

$$\mathit{bit}_i^T \sqsubseteq \exists S^-, \qquad \text{for } 1 \leq i < k, \tag{53}$$

$$\mathit{bit}_k^T \sqsubseteq \neg \exists S^-, \qquad \mathit{bit}_{k+i}^T \sqsubseteq \begin{cases} \neg \exists S^-, & \text{if } i = \mathit{left}(T), \\ \exists S^-, & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq i \leq k, \tag{54}$$

$$\mathit{bit}_{2k+1}^T \sqsubseteq \neg \exists S^-, \qquad \mathit{bit}_{2k+1+i}^T \sqsubseteq \begin{cases} \neg \exists S^-, & \text{if } i = \mathit{down}(T), \\ \exists S^-, & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq i \leq k, \tag{55}$$

$$\mathit{bit}_{3k+2}^T \sqsubseteq \neg \exists S^-, \qquad \mathit{bit}_{3k+2+i}^T \sqsubseteq \begin{cases} \neg \exists S^-, & \text{if } i = \mathit{up}(T), \\ \exists S^-, & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq i \leq k, \tag{56}$$

$$\mathit{bit}_{4k+3}^T \sqsubseteq \exists S^-, \qquad \mathit{bit}_{4k+3+i}^T \sqsubseteq \neg \exists S^-, \qquad \text{for } 1 \leq i \leq k, \tag{57}$$

$$\mathit{bit}_{5k+4}^T \sqsubseteq \exists S^-, \qquad \mathit{bit}_{5k+4+i}^T \sqsubseteq \begin{cases} \exists S^-, & \text{if } i = \mathit{right}(T), \\ \neg \exists S^-, & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq i \leq k. \tag{58}$$

Suppose that a domain element w is an instance of T at some moment t, for some $T \in \mathfrak{T}$. Then, w will be an instance of $\exists S^-$ at the moments $t, \dots, t+k-1$. We think of this time interval on w (and, as before, on z_0) as the *plug*, or the *P-section*. After the plug, we have a one-instant *gap* (where w is an instance of $\neg \exists S^-$). The gap is followed by a sequence of k moments of time that represent *left*(T) in the sense that only at the ith moment of the sequence, where $i = \mathit{left}(T)$, w does *not* have an S-predecessor. Then, we again have a one-instant gap, followed by a sequence of k-moments representing *down*(T) (in the same sense), another one-instant gap, and a sequence representing *up*(T) (Figure 9). At the next moment, $t+4k+3$, w will be an instance of $\exists S^-$; then, we have k gaps (i.e., $\neg \exists S^-$), called the *socket*, or the *S-section*. After the socket, at $t+5k+4$, w is again an instance of $\exists S^-$, and then we have a sequence of k moments representing "inverted"

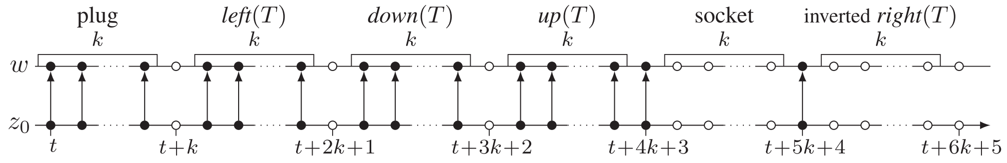

Fig. 9. Representing a tile using an S-successor.

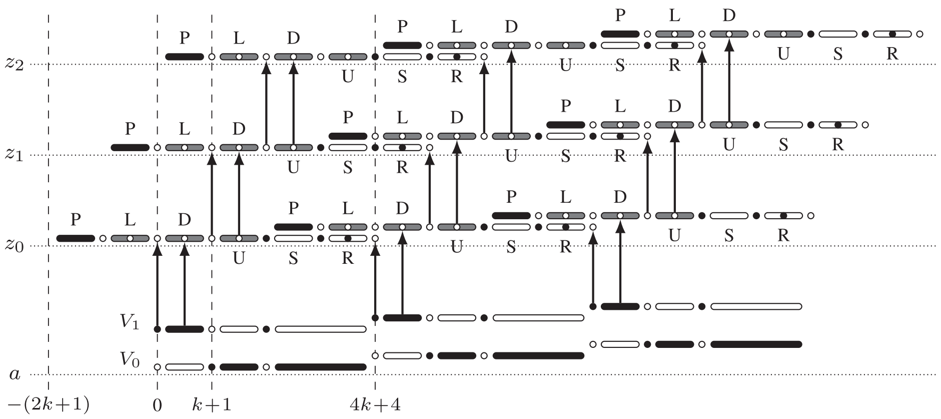

Fig. 10. The structure of a model of $\mathcal{K}_{\mathfrak{T}}$.

right(T): the ith moment of this sequence has an S-predecessor iff $i = right(T)$. We note that by (50), the pattern of $\exists S^-$ on w in Figure 9 is reflected by the pattern of $\exists S$ on the S-predecessor z_0 of w at t, which (partly) justifies our terminology when we say that *tile T is placed on* z_0 (rather than on w).

Thus, if the preceding concept inclusions hold, a tile—denote it by T_{00}—is placed on z_0 at the moment $-(2k+1)$, or, equivalently, T_{00} is placed on an S-successor w of z_0. The following concept inclusions will ensure then that the tiling is extended properly along both axes:

$$\exists R^- \sqcap \Box_P \neg\exists R^- \sqsubseteq \Box_F(\exists S \sqcup \exists R \sqcup \exists R^-), \tag{59}$$

$$\exists R^- \sqcap \Box_P \neg\exists R^- \sqsubseteq \Box_P \neg\exists R, \tag{60}$$

$$\exists V_0 \sqcap \exists R \sqsubseteq \bot, \tag{61}$$

$$\exists V_0 \sqcap \exists R^- \sqsubseteq \bot, \tag{62}$$

$$\exists S \sqcap \exists R \sqsubseteq \bot, \tag{63}$$

$$\exists S \sqcap \exists R^- \sqsubseteq \bot, \tag{64}$$

$$\exists R \sqcap \exists R^- \sqsubseteq \bot. \tag{65}$$

Indeed, consider the elements z_0 and w with the tile T_{00} placed on z_0 at $-(2k+1)$. Then, w has gaps (i.e., no incoming S-arrows) at moments 0, $down(T_{00})$, $k+1$, $k+1+up(T_{00})$, k gaps from $2k+3$ to $3k+2$ and $k-1$ gaps from $3k+4$ to $4k+3$ (one of the positions is not a gap because of the inverted representation of $right(T_{00})$). By (59), each of those positions on z_0 must be filled either by an outgoing S-arrow, or by an incoming R-arrow, or by an outgoing R-arrow. Consider now what happens in these positions (Figure 10).

(1) We know that there is an incoming R-arrow at 0 (i.e., z_0 is an instance of $\exists R^-$), and so, by (64) and (65), z_0 cannot be an instance of $\exists S$ and $\exists R$ at 0.

(2) The position at $down(T_{00})$ is filled by an incoming R-arrow using the following concept inclusions (by (64), the incoming R-arrow can only appear at $down(T_{00})$):

$$A \sqsubseteq \bigsqcup_{1 \leq i \leq k} \Diamond_F^{=i}\, \mathit{init\text{-}bot}, \tag{66}$$

$$\mathit{init\text{-}bot} \sqsubseteq \exists R. \tag{67}$$

(3) The position at $k+1$ cannot be filled by an outgoing S-arrow, because that would trigger a new tile sequence, which would require k S-arrows of the P-section, which is impossible due to (51). Next, as we observed earlier, $a^{\mathcal{I}}$ belongs to $\exists V_0$ at all moments i with $k+1 \leq i \leq 2k+1$, and so, by (28) and (61), z_0 cannot have an incoming R-arrow at moment $k+1$. Thus, the position at $k+1$ must be filled by an outgoing R-arrow. Thus, there is an R-successor z_1 of z_0, which, by (28), implies that z_1 has no incoming R-arrows before $k+1$. Then, by (47) through (58), there will be a tile placed on z_1 at $-k = (k+1) - (2k+1)$.

(4) Similarly, the position at $k+1+up(T)$ must be filled by an outgoing R-arrow, which ensures that the *down*-colour of the tile placed on z_1 matches the *up*-colour of the tile on z_0.

(5) The k positions of the S-section from $2k+3$ to $3k+2$ cannot be filled by an incoming R-arrow. On the other hand, the tile placed on z_1 has its *up*-colour encoded in this range, and so an outgoing R-arrow cannot fill *all* of these gaps either (as $k > 1$). So, z_0 has another S-successor x_1 in at least one of the moments of the S-section. By (48) and (50), x_1 does not belong to $\exists S^-$ before $-(2k+1)$. By (49), a tile is placed on x_1 between $-(2k+1)$ and $3k+2$, but, by (51) and because the tile requires x_1 to be the S-successor for k consecutive moments of the P-section, it is only possible at $2k+2$. Moreover, since the *left*- and *right*-sections of these tile sequences overlap on z_0, by (51), the adjacent colours of these two tiles match. This ensures that the $k-1$ gaps of the inverted representation of the *right*-colour of the first tile are also filled.

Let $\mathcal{K}_{\mathfrak{T}}$ be the KB containing all preceding concept inclusions and $\mathcal{A}$. If $\mathcal{I}$ is a model of $\mathcal{K}_{\mathfrak{T}}$, then the process described earlier generates a sequence $z_0, z_1, \dots$ of domain elements such that each z_i has a tile placed on it at every $4k+4$ moments of time; moreover, the R-arrows form a proper $\mathbb{N} \times \mathbb{N}$-grid and the adjacent colours of the tiles match. We only note that the gaps at positions in the *down*-section do not need a special treatment after the very first tile T_{00} at $(0,0)$, because for each z_i, the sequence of tiles on z_{i+1} will have their *left*- and *right*-sections, with no gap to be filled by an incoming R-arrow; thus, the only available choice for tiles on z_i is $\exists R$.

We have proved that if $\mathcal{K}_{\mathfrak{T}}$ is satisfiable, then $\mathfrak{T}$ tiles $\mathbb{N} \times \mathbb{N}$. The converse implication is shown using the satisfying interpretation illustrated in Figure 10. □

6.2. Undirected Temporal Operators: Decidability and NP-completeness

If we disallow the “previous time,” “next time,” “always in the past,” and “always in the future” operators in the language of concept inclusions and replace them with “always” ($\boxast$), then reasoning in the resulting logic $T_U^*DL\text{-}Lite_{bool}^{\mathcal{N}}$ becomes decidable and NP-complete.

Obviously, the problem is NP-hard (because of the underlying DL). However, rather surprisingly, the interaction of temporalised roles and number restrictions is yet another source of nondeterminism, which is exhibited already by very simple TBoxes with concept inclusions in the *core* fragment. The following example illustrates this point and gives a glimpse of the difficulties that we shall face in the proof of the NP upper bound by means of the quasimodel technique: unlike other quasimodel proofs [Gabbay et al. 2003], where only types of domain elements need to be guessed, here we also have to guess relations between ABox individuals at all relevant moments of time.

Example 6.2. Let $\mathcal{T} = \{\, A \sqsubseteq\, \geq 5\, R,\ \geq 7 \diamondast R \sqsubseteq \bot \,\}$ and

$$\mathcal{A} = \{\, \bigcirc_F A(a),\ R(a, b_1),\ R(a, b_2),\ R(a, b_3),\ \bigcirc_F R(a, b_1) \,\}.$$

The second concept inclusion of the TBox implies that in any every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$, a cannot have more than six $(\diamondast R)^{\mathcal{I}}$-successors in total; thus, it can only have b_1, b_2, b_3, and up to three $R^{\mathcal{I}}$-successors from outside the ABox. At moment 1, however, a must have at least five $R^{\mathcal{I}}$-successors, including b_1. Thus, one of the $(\diamondast R)^{\mathcal{I}}$-successors in the ABox has to be reused: we have either $\mathcal{I} \models \bigcirc_F R(a, b_2)$ or $\mathcal{I} \models \bigcirc_F R(a, b_3)$.

Consider now $\mathcal{T} = \{\, \geq 6 \diamondast R \sqsubseteq \bot,\ \top \sqsubseteq\, \geq 4 \boxast R \,\}$ and $\mathcal{A} = \{\, R(a, b_1), R(a, b_2) \,\}$. Then, in every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$, either $\mathcal{I} \models \boxast R(a, b_1)$ or $\mathcal{I} \models \boxast R(a, b_2)$.

THEOREM 6.3. *The satisfiability problem for T^{*}_{U}DL-Lite$^{\mathcal{N}}_{bool}$ KBs is* NP-*complete.*

PROOF. Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a T^{*}_{U}DL-Lite$^{\mathcal{N}}_{bool}$ KB. In what follows, given an interpretation $\mathcal{I}$, we write $(\geq q \boxast R)^{\mathcal{I}}$ and $(\geq q \diamondast R)^{\mathcal{I}}$ instead of $(\geq q \boxast R)^{\mathcal{I}(n)}$ and $(\geq q \diamondast R)^{\mathcal{I}(n)}$, for $n \in \mathbb{Z}$, because temporalised roles are time invariant. As before, $\mathsf{ob}_{\mathcal{A}}$ denotes the set of all object names occurring in $\mathcal{A}$ (we assume $|\mathsf{ob}_{\mathcal{A}}| \geq 1$) and $\mathsf{role}_{\mathcal{K}}$ the set of role names in $\mathcal{K}$ and their inverses. Let $Q_{\mathcal{T}} \subseteq \mathbb{N}$ be the set (cf. p. 19) comprised of 1 and all q such that one of $\geq q \boxast R$, $\geq q\, R$, or $\geq q \diamondast R$ occurs in $\mathcal{T}$, and let $Q_{\mathcal{A}}$ be the set of all natural numbers from 0 to $|\mathsf{ob}_{\mathcal{A}}|$. Let $q_{\mathcal{K}} = \max(Q_{\mathcal{T}} \cup Q_{\mathcal{A}}) + 1$. First, we show that it is enough to consider interpretations with the number of $\boxast R$-successors bounded by $q_{\mathcal{K}} - 1$ (see Appendix C for a proof):

LEMMA 6.4. *Every satisfiable T^{*}_{U}DL-Lite$^{\mathcal{N}}_{bool}$ KB $\mathcal{K}$ can be satisfied in an interpretation $\mathcal{I}$ with $(\geq q_{\mathcal{K}} \boxast R)^{\mathcal{I}} = \emptyset$, for each $R \in \mathsf{role}_{\mathcal{K}}$.*

Next, we define the notion of quasimodel. Let $Q \supseteq Q_{\mathcal{T}} \cup Q_{\mathcal{A}}$ be a set of natural numbers with $\max Q = q_{\mathcal{K}} - 1$. We assume that the usual order on the natural numbers in $\widehat{Q} = Q \cup \{\omega\}$ is extended to ω, which is the greatest element: $0 < 1 < \cdots < q_{\mathcal{K}} - 1 < \omega$. Let Σ consist of the following concepts and their negations: subconcepts of concepts occurring in $\mathcal{K}$ and $\geq q \boxast R$, $\geq q\, R$, and $\geq q \diamondast R$, for all $R \in \mathsf{role}_{\mathcal{K}}$ and $q \in \widehat{Q}$. A Σ-*type* $\boldsymbol{t}$ is a maximal consistent subset of Σ:

(t₁) $C \in \boldsymbol{t}$ iff $\neg C \notin \boldsymbol{t}$, for each $C \in \Sigma$,
(t₂) $C_1 \sqcap C_2 \in \boldsymbol{t}$ iff $C_1, C_2 \in \boldsymbol{t}$, for each $C_1 \sqcap C_2 \in \Sigma$,
(t₃) if $\boxast C \in \boldsymbol{t}$ then $C \in \boldsymbol{t}$, for each $\boxast C \in \Sigma$,
(t₄) if $\geq q\, R \in \boldsymbol{t}$, then $\geq q'\, R \in \boldsymbol{t}$, for each $\geq q'\, R \in \Sigma$ with $q > q'$ (similarly for $\boxast R$ and $\diamondast R$),
(t₅) $\geq 0 \boxast R \in \boldsymbol{t}$ but $\geq \omega \boxast R \notin \boldsymbol{t}$, for each role R,
(t₆) if $\geq q \boxast R \in \boldsymbol{t}$, then $\geq q\, R \in \boldsymbol{t}$ and if $\geq q\, R \in \boldsymbol{t}$ then $\geq q \diamondast R \in \boldsymbol{t}$, for each role R,
(t₇) if $\geq q \diamondast R \in \boldsymbol{t}$, then $\geq q \boxast R \in \boldsymbol{t}$, for each *rigid* role R.

Denote by $Z_{\mathcal{A}}$ the set of all integers k such that at least one of $\bigcirc^k A(a)$, $\bigcirc^k \neg A(a)$, $\bigcirc^k S(a, b)$, or $\bigcirc^k \neg S(a, b)$ occurs in $\mathcal{A}$, and let $Z \supseteq Z_{\mathcal{A}}$ be a finite set of integers.

By a (Z, Σ)-*run* (or simply run, if Z and Σ are clear from the context), we mean a function r from Z to the set of Σ-types. Concepts of the form $\boxast C$, $\geq q \boxast R$, $\geq q \diamondast R$, and their negations are called *rigid*. A run r is said to be *coherent* if the following holds for each rigid concept D in Σ:

(r₁) if $D \in r(k_0)$, for some $k_0 \in Z$, then $D \in r(k)$ for all $k \in Z$.

In the following, the runs are assumed to be coherent, and so, for rigid concepts D, we can write $D \in r$ in place of $D \in r(k)$, for some (all) $k \in Z$. The *required R-rank of r at*

$k \in Z$ and the *required* $\boxast R$- and $\diamondast R$-*ranks of* r are defined by taking

$$\varrho_r^{R,k} = \max\{q \in \widehat{Q} \mid {\geq q}\, R \in r(k)\}, \qquad \begin{aligned}\varrho_r^{\Box R} &= \max\{q \in \widehat{Q} \mid {\geq q} \boxast R \in r\},\\ \varrho_r^{\Diamond R} &= \max\{q \in \widehat{Q} \mid {\geq q} \diamondast R \in r\}.\end{aligned}$$

By the definition of Σ-types, $\varrho_r^{\Box R} \leq \varrho_r^{R,k} \leq \varrho_r^{\Diamond R}$. Moreover, if R is rigid then $\varrho_r^{\Box R} = \varrho_r^{\Diamond R} < \omega$. For flexible roles, however, the inequalities may be strict. A run r is *saturated* if the following hold:

(r$_2$) for every flexible role $R \in \mathsf{role}_\mathcal{K}$, if $\varrho_r^{\Box R} < \varrho_r^{\Diamond R}$, then
- —there is $k_0 \in Z$ with $\varrho_r^{\Box R} < \varrho_r^{R,k_0}$, and
- —if additionally $\varrho_r^{\Diamond R} < \omega$, then there is $k_1 \in Z$ with $\varrho_r^{R,k_1} < \varrho_r^{\Diamond R}$;

(r$_3$) for every $\boxast C \notin r$, there is $k_0 \in Z$ with $C \notin r(k_0)$.

Finally, we call $\mathcal{E}$ a *consistent Z-extension of* $\mathcal{A}$ if $\mathcal{E}$ extends $\mathcal{A}$ with assertions of the form $\bigcirc^k S(a,b)$, for $k \in Z$ and $a, b \in \mathsf{ob}_\mathcal{A}$, such that $\bigcirc^k S(a,b) \notin \mathcal{E}$ for all $\bigcirc^k \neg S(a,b) \in \mathcal{A}$. Example 6.2 shows that given an ABox, we have first to guess such an extension to describe a quasimodel. More precisely, we have to count the number of R-successors among the ABox individuals in $\mathcal{E}$. To this end, define the following sets, for $a \in \mathsf{ob}_\mathcal{A}$, $R \in \mathsf{role}_\mathcal{K}$, and $k \in Z$:

$$\mathcal{E}_a^{R,k} = \{b \mid \bigcirc^k R(a,b) \in \mathcal{E}\}, \qquad \begin{aligned}\mathcal{E}_a^{\Box R} &= \{b \mid \bigcirc^n R(a,b) \in \mathcal{E},\ \text{for all } n \in Z\},\\ \mathcal{E}_a^{\Diamond R} &= \{b \mid \bigcirc^n R(a,b) \in \mathcal{E},\ \text{for some } n \in Z\},\end{aligned}$$

where, as on p. 19, we assume that $\mathcal{E}$ contains $\bigcirc^n S^-(b,a)$ whenever it contains $\bigcirc^n S(a,b)$. We say that a (Z, Σ)-run r is *a-faithful for* $\mathcal{E}$ if

(r$_4$) $A \in r(k)$, for all $\bigcirc^k A(a) \in \mathcal{E}$, and $\neg A \in r(k)$, for all $\bigcirc^k \neg A(a) \in \mathcal{E}$;
(r$_5$) $0 \leq \varrho_r^{\Box R} - |\mathcal{E}_a^{\Box R}| \leq \varrho_r^{R,k} - |\mathcal{E}_a^{R,k}| \leq \varrho_r^{\Diamond R} - |\mathcal{E}_a^{\Diamond R}|$, for all $R \in \mathsf{role}_\mathcal{K}$ and $k \in Z$;[7]
(r$_6$) for all $R \in \mathsf{role}_\mathcal{K}$, if $\varrho_r^{\Box R} - |\mathcal{E}_a^{\Box R}| < \varrho_r^{\Diamond R} - |\mathcal{E}_a^{\Diamond R}|$, then
- —there is $k_0 \in Z$ with $\varrho_r^{\Box R} - |\mathcal{E}_a^{\Box R}| < \varrho_r^{R,k_0} - |\mathcal{E}_a^{R,k_0}|$, and
- —if additionally $\varrho_r^{\Diamond R} < \omega$, then there is $k_1 \in Z$ with $\varrho_r^{R,k_1} - |\mathcal{E}_a^{R,k_1}| < \varrho_r^{\Diamond R} - |\mathcal{E}_a^{\Diamond R}|$.

Condition **(r$_5$)** says that the number of R-successors in the ABox extension $\mathcal{E}$ does not exceed the number of required R-successors: in view of $|\mathcal{E}_a^{R,k}| \in Q_\mathcal{A} \subseteq Q$, we have ${\geq q}\, R \in r(k)$ for $q = |\mathcal{E}_a^{R,k}|$ (and similarly for $\boxast R$ and $\diamondast R$). Condition **(r$_5$)** also guarantees that the number of required R-successors that are *not* ABox individuals is consistent for $\boxast R$, R, and $\diamondast R$. In particular, since $|\mathcal{E}_a^{\Box R}| \leq |\mathcal{E}_a^{\Diamond R}|$, it follows that $\varrho_r^{\Box R} = \varrho_r^{\Diamond R}$ implies $|\mathcal{E}_a^{\Box R}| = |\mathcal{E}_a^{\Diamond R}|$, and so $\bigcirc^n R(a,b) \in \mathcal{E}$, for all $n \in Z$, whenever $\bigcirc^k R(a,b) \in \mathcal{E}$ for some $k \in Z$. Finally, **(r$_6$)** is an adaptation of the notion of saturated runs for the case of ABox individuals.

A *quasimodel* $\mathfrak{Q}$ *for* $\mathcal{K}$ is a quadruple $(Q, Z, \mathfrak{R}, \mathcal{E})$, where Q and Z are finite sets of integers extending $Q_\mathcal{T} \cup Q_\mathcal{A}$ and $Z_\mathcal{A}$, respectively, $\mathfrak{R}$ is a set of coherent and saturated (Z, Σ)-runs (for Σ defined on the basis of $\widehat{Q}$), and $\mathcal{E}$ is a consistent Z-extension of $\mathcal{A}$ satisfying the following conditions:

(Q$_1$) for all $r \in \mathfrak{R}$, $k \in Z$, and $C_1 \sqsubseteq C_2 \in \mathcal{T}$, if $C_1 \in r(k)$, then $C_2 \in r(k)$;
(Q$_2$) for all $a \in \mathsf{ob}_\mathcal{A}$, there is a run $r_a \in \mathfrak{R}$ that is a-faithful for $\mathcal{E}$;
(Q$_3$) for all $R \in \mathsf{role}_\mathcal{K}$, if there is $r \in \mathfrak{R}$ with $\varrho_r^{\Box R} \geq 1$, then there is $r' \in \mathfrak{R}$ with $\varrho_{r'}^{\Box \mathsf{inv}(R)} \geq 1$;

[7]We assume that $\omega - n = \omega$ for any natural number n.

($\mathbf{Q_4}$) for all $R \in \mathsf{role}_{\mathcal{K}}$, if there is $r \in \mathfrak{R}$ with $\varrho_r^{\Box R} < \varrho_r^{\Diamond R}$, then there exists $r' \in \mathfrak{R}$ with $\varrho_{r'}^{\Box \mathsf{inv}\,(R)} < \varrho_{r'}^{\Diamond \mathsf{inv}\,(R)}$.

Condition **($\mathbf{Q_1}$)** ensures that all runs are consistent with the concept inclusions in $\mathcal{T}$ and **($\mathbf{Q_2}$)** that there are runs for all ABox individuals; **($\mathbf{Q_3}$)** guarantees that a $\boxast\, R$-successor can be found whenever required; and **($\mathbf{Q_4}$)** provides R- (and thus $\diamondast R$-) successors whenever required. The following lemma states that the notion of quasimodel is adequate for checking satisfiability of $T_U^{*}DL\text{-}Lite_{bool}^{\mathcal{N}}$ KBs:

Lemma 6.5. *A $T_U^{*}DL\text{-}Lite_{bool}^{\mathcal{N}}$ KB $\mathcal{K}$ is satisfiable iff there is a quasimodel $\mathfrak{Q}$ for $\mathcal{K}$ such that the size of $\mathfrak{Q}$ is polynomial in the size of $\mathcal{K}$.*

Proof. ($\Rightarrow$) Let $\mathcal{I}$ be a model of $\mathcal{K}$. By Lemma 6.4, we may assume that for each $R \in \mathsf{role}_{\mathcal{K}}$, the number of $\boxast\, R$-successors of any element in $\mathcal{I}$ does not exceed $q_{\mathcal{K}} - 1$. We construct a polynomial-size quasimodel $\mathfrak{Q} = (Q, Z, \mathfrak{R}, \mathcal{E})$ for $\mathcal{K}$. First, we select a set D of elements of $\Delta^{\mathcal{I}}$ that serve as prototypes for runs in $\mathfrak{R}$: each $u \in D$ will give rise to a run r_u in $\mathfrak{R}$ (after the set Z of time instants has been fixed). Set $D_0 = \{a^{\mathcal{I}} \mid a \in \mathsf{ob}_{\mathcal{A}}\}$ and then proceed by induction: if D_m has already been defined, then we construct D_{m+1} by extending D_m as follows:

—if $D_m \cap (\exists \boxast\, R)^{\mathcal{I}} \neq \emptyset$ but $D_m \cap (\exists \boxast\, \mathsf{inv}\,(R))^{\mathcal{I}} = \emptyset$, then we add some $u \in (\exists \boxast\, \mathsf{inv}\,(R))^{\mathcal{I}}$;
—if there is q with $D_m \cap ((\geq q \diamondast R)^{\mathcal{I}} \setminus (\geq q \boxast\, R)^{\mathcal{I}}) \neq \emptyset$ but there is no q' with $D_m \cap ((\geq q' \diamondast \mathsf{inv}\,(R))^{\mathcal{I}} \setminus (\geq q' \boxast\, \mathsf{inv}\,(R))^{\mathcal{I}}) \neq \emptyset$, then add $u \in (\geq q'' \diamondast \mathsf{inv}\,(R))^{\mathcal{I}} \setminus (\geq q'' \boxast\, \mathsf{inv}\,(R))^{\mathcal{I}}$ for some q'' (by Lemma 6.4, we assume that q, q', and q'' do not exceed $q_{\mathcal{K}}$).

When neither rule is applicable to D_m, stop and set $D = D_m$. Clearly, we have $|D| \leq |\mathsf{ob}_{\mathcal{A}}| + 2|\mathsf{role}_{\mathcal{K}}|$.

For each $u \in D$, let

$$\rho_u^{\Box R} = \max\{q < q_{\mathcal{K}} \mid u \in (\geq q \boxast\, R)^{\mathcal{I}}\},$$
$$\rho_u^{\Diamond R} = \begin{cases} \omega, & \text{if } u \in (\geq q_{\mathcal{K}} \diamondast R)^{\mathcal{I}}, \\ \max\{q < q_{\mathcal{K}} \mid u \in (\geq q \diamondast R)^{\mathcal{I}}\}, & \text{otherwise.} \end{cases}$$

We now choose time instants to be included in the runs $\mathfrak{R}$. Let Z extend $Z_{\mathcal{A}}$ with the following:

($\mathbf{Z_0}$) for any $u \in D$ and $\boxast\, C \in \Sigma$ such that $u \notin (\boxast\, C)^{\mathcal{I}}$, we add some $n \in \mathbb{Z}$ with $u \notin C^{\mathcal{I}(n)}$;

($\mathbf{Z_1}$) for any $u \in D$ and $R \in \mathsf{role}_{\mathcal{K}}$ such that $\rho_u^{\Box R} < \rho_u^{\Diamond R}$, we add

—some $n_0 \in \mathbb{Z}$ with $u \in (\geq (\rho_u^{\Box R} + 1)\, R)^{\mathcal{I}(n_0)}$ and
—if additionally $\rho_u^{\Diamond R} < \omega$, some $n_1 \in \mathbb{Z}$ with $u \notin (\geq \rho_u^{\Diamond R}\, R)^{\mathcal{I}(n_1)}$;

($\mathbf{Z_2}$) for any $a, b \in \mathsf{ob}_{\mathcal{A}}$ and $R \in \mathsf{role}_{\mathcal{K}}$ such that $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in (\diamondast R)^{\mathcal{I}}$, we add

—some $n_0 \in \mathbb{Z}$ with $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}(n_0)}$ and
—if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin (\boxast\, R)^{\mathcal{I}}$, some $n_1 \in \mathbb{Z}$ with $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin R^{\mathcal{I}(n_1)}$;

($\mathbf{Z_3}$) for any $a \in \mathsf{ob}_{\mathcal{A}}$ and $R \in \mathsf{role}_{\mathcal{K}}$ such that $\rho_{a^{\mathcal{I}}}^{\Box R} - |\mathcal{I}_a^{\Box R}| < \rho_{a^{\mathcal{I}}}^{\Diamond R} - |\mathcal{I}_a^{\Diamond R}|$, we add

—some $n_0 \in \mathbb{Z}$ with $a^{\mathcal{I}} \in (\geq (q_0 + 1)\, R)^{\mathcal{I}(n_0)}$, for $q_0 = \rho_{a^{\mathcal{I}}}^{\Box R} + (|\mathcal{I}_a^{R,n_0}| - |\mathcal{I}_a^{\Box R}|)$, and
—if $\rho_{a^{\mathcal{I}}}^{\Diamond R} < \omega$, some $n_1 \in \mathbb{Z}$ with $a^{\mathcal{I}} \notin (\geq q_1\, R)^{\mathcal{I}(n_1)}$, for $q_1 = \rho_{a^{\mathcal{I}}}^{\Diamond R} - (|\mathcal{I}_a^{\Diamond R}| - |\mathcal{I}_a^{R,n_1}|)$,

where $\mathcal{I}_a^{R,k} = \{b \in \mathsf{ob}_{\mathcal{A}} \mid (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}(k)}\}$ and $\mathcal{I}_a^{\Box R}$ and $\mathcal{I}_a^{\Diamond R}$ are defined similarly.

Clearly, $|Z_0| \leq |D| \cdot |\mathcal{K}|$, $|Z_1| \leq 2|D| \cdot |\mathsf{role}_\mathcal{K}|$, $|Z_2| \leq 2|\mathsf{ob}_\mathcal{A}|^2 \cdot |\mathsf{role}_\mathcal{K}|$ and $|Z_3| \leq 2 \cdot |\mathsf{ob}_\mathcal{A}| \cdot |\mathsf{role}_\mathcal{K}|$. Thus, $|Z| = O(|\mathcal{K}|^3)$. The time instants in Z_0, Z_1, and Z_2 exist because $\mathcal{I} \models \mathcal{K}$. We now show that n_0 required in Z_3 also exists. Suppose, on the contrary, that $a^\mathcal{I} \notin (\geq (q_0 + 1)\, R)^{\mathcal{I}(n)}$, with q_0 as earlier, for all $n \in \mathbb{Z}$. Then, $a^\mathcal{I}$ has at most $(\rho^{\Box R}_{a^\mathcal{I}} + (|\mathcal{I}^{R,n}_a| - |\mathcal{I}^{\Box R}_a|))$-many R-successors, whence the number of non-ABox R-successors of $a^\mathcal{I}$ does not exceed $\rho^{\Box R}_{a^\mathcal{I}} - |\mathcal{I}^{\Box R}_a|$. So, at all instants $n \in \mathbb{Z}$, every R-successor of $a^\mathcal{I}$ is either in $\mathsf{ob}_\mathcal{A}$ or is in fact a $\boxast$ R-successor, contrary to $\rho^{\Box R}_{a^\mathcal{I}} - |\mathcal{I}^{\Box R}_a| < \rho^{\Diamond R}_{a^\mathcal{I}} - |\mathcal{I}^{\Diamond R}_a|$. Using a similar argument, one can show that n_1 required in Z_3 exists as well. Having fixed Z, we define a consistent Z-extension $\mathcal{E}$ of $\mathcal{A}$ by taking

$$\mathcal{E} = \mathcal{A} \cup \{\bigcirc^k S(a, b) \;\mid\; (a^\mathcal{I}, b^\mathcal{I}) \in S^{\mathcal{I}(k)} \text{ and } k \in Z\}.$$

Let Q be the set comprising $Q_\mathcal{T}$, $Q_\mathcal{A}$ and, for any $u \in D$ and $R \in \mathsf{role}_\mathcal{K}$, the integers from

$$\rho^{\Box R}_u, \;\; \rho^{\Diamond R}_u \quad \text{and} \quad \max\{q < q_\mathcal{K} \mid u \in (\geq q\, R)^{\mathcal{I}(k)}\}, \text{ for } k \in Z.$$

By definition, $\max Q = q_\mathcal{K} - 1$ and $|Q| \leq |Q_\mathcal{T}| + |Q_\mathcal{A}| + |D| \cdot |\mathsf{role}_\mathcal{K}| \cdot (2 + |Z|)$. Let $\mathfrak{R}$ be the set of (Z, Σ)-runs r_u, for $u \in D$, defined by taking, for each $k \in Z$,

—$\geq \omega\, R \in r_u(k)$ iff $u \in (\geq q_\mathcal{K}\, R)^{\mathcal{I}(k)}$, and $\geq \omega \diamondast R \in r_u(k)$ iff $u \in (\geq q_\mathcal{K} \diamondast R)^\mathcal{I}$,
—$C \in r_u(k)$ iff $u \in C^{\mathcal{I}(k)}$, for all other concepts $C \in \Sigma$.

Since $\mathcal{I} \models \mathcal{K}$ and $\mathcal{I}$ is as in Lemma 6.4, each $r_u(k)$ is a Σ-type. Each $r_u \in \mathfrak{R}$ is a coherent and saturated (Z, Σ)-run: $(\mathbf{r_1})$ holds because $\mathcal{I} \models \mathcal{K}$; $(\mathbf{r_3})$ and $(\mathbf{r_2})$ are due to $Z_0 \subseteq Z$ and $Z_1 \subseteq Z$, respectively. Since $\mathcal{I} \models \mathcal{E}$, each run $r_{a^\mathcal{I}}$ is a-faithful for $\mathcal{E}$. Indeed, $(\mathbf{r_4})$ is due to $Z_\mathcal{A} \subseteq Z$. To show $(\mathbf{r_5})$ and $(\mathbf{r_6})$, observe that by definition, $|\mathcal{E}^{R,k}_a| = |\mathcal{I}^{R,k}_a|$ and, since $Z_2 \subseteq Z$, we also have $|\mathcal{E}^{\Box R}_a| = |\mathcal{I}^{\Box R}_a|$ and $|\mathcal{E}^{\Diamond R}_a| = |\mathcal{I}^{\Diamond R}_a|$; moreover, $\varrho^{\Box R}_{r_{a^\mathcal{I}}}, \varrho^{R,k}_{r_{a^\mathcal{I}}}, \varrho^{\Diamond R}_{r_{a^\mathcal{I}}} \in \widehat{Q}$ and, for each $q < q_\mathcal{K}$, we have $a^\mathcal{I} \in (\geq q\, R)^{\mathcal{I}(k)}$ iff $\varrho^{R,k}_{r_{a^\mathcal{I}}} \geq q$ (and similarly for $\boxast$ R and $\diamondast$ R). Then, $(\mathbf{r_5})$ follows from the choice of $\mathcal{E}$ and $(\mathbf{r_6})$ from $Z_3 \subseteq Z$. We claim that $\mathfrak{Q} = (Q, Z, \mathfrak{R}, \mathcal{E})$ is a quasimodel for $\mathcal{K}$: $(\mathbf{Q_1})$ holds by definition and $(\mathbf{Q_2})$ through $(\mathbf{Q_4})$ follow from the choice of D. Finally, as $|\mathcal{E}| \leq |\mathcal{A}| + |Z| \cdot |\mathsf{ob}_\mathcal{A}|^2 \cdot |\mathsf{role}_\mathcal{K}|$ and $|\mathfrak{R}| \leq |\mathsf{ob}_\mathcal{A}| + 2|\mathsf{role}_\mathcal{K}|$, the quasimodel is of polynomial size.

($\Leftarrow$) Let $\mathfrak{Q} = (Q, Z, \mathfrak{R}, \mathcal{E})$ be a quasimodel for $\mathcal{K}$. We construct an interpretation $\mathcal{I}$ satisfying $\mathcal{K}$, which is based on some domain $\Delta^\mathcal{I}$ that will be defined inductively as the union

$$\Delta^\mathcal{I} = \bigcup\nolimits_{m \geq 0} \Delta_m, \qquad \text{where } \Delta_m \subseteq \Delta_{m+1}, \text{ for all } m \geq 0.$$

Each set Δ_{m+1} ($m \geq 0$) is constructed by adding to Δ_m new domain elements that are copied from the runs in $\mathfrak{R}$; similarly to the proof of Theorem 4.1 in Appendix A, the function $cp : \Delta^\mathcal{I} \to \mathfrak{R}$ keeps track of this process. In contrast to the proof of Theorem 4.1, however, the runs are defined on a finite set, Z, and so we need to multiply (and rearrange) the time instants of Z when creating elements of $\Delta^\mathcal{I}$ from runs in $\mathfrak{R}$. To this end, for each $u \in \Delta^\mathcal{I}$, we define a function $\nu_u : \mathbb{Z} \to Z$ that maps each time instant $n \in \mathbb{Z}$ of $u \in \Delta^\mathcal{I}$ to its "origin" $\nu_u(n) \in Z$ on the run $cp(u)$. Since the constructed interpretation $\mathcal{I}$ may contain infinite sequences of domain elements related by roles, we will need to ensure that each Σ-type of the run appears infinitely often along $\mathbb{Z}$ (note, however, that the actual order of time instants is important only for $Z_\mathcal{A}$, the instants of the ABox).

The interpretation of role names in $\mathcal{I}$ is constructed inductively along with the construction of the domain: $S^{\mathcal{I}(n)} = \bigcup_{m \geq 0} S^{n,m}$, where $S^{n,m} \subseteq \Delta_m \times \Delta_m$, for $m \geq 0$. Given $m \geq 0$ and $u \in \Delta_m$, we define the *actual S-rank at moment $n \in \mathbb{Z}$* and the *actual*

$\boxast$ *S- and* $\diamondast$ *S-ranks on step m*:

$$\tau_{u,m}^{S,n} = \sharp\{u' \in \Delta_m \mid (u,u') \in S^{n,m}\}, \qquad \begin{aligned} \tau_{u,m}^{\Box S} &= \sharp\{u' \in \Delta_m \mid (u,u') \in S^{k,m}, \text{ for all } k \in \mathbb{Z}\},\\ \tau_{u,m}^{\Diamond S} &= \sharp\{u' \in \Delta_m \mid (u,u') \in S^{k,m}, \text{ for some } k \in \mathbb{Z}\}.\end{aligned}$$

The actual S^--, $\boxast S^-$-, and $\diamondast S^-$-ranks are defined similarly, with (u,u') replaced by (u',u). Let

$$\delta_{u,m}^{\Box R} = \varrho_{cp(u)}^{\Box R} - \tau_{u,m}^{\Box R}, \qquad \delta_{u,m}^{R,n} = \varrho_{cp(u)}^{R,\nu_u(n)} - \tau_{u,m}^{R,n}, \quad \text{and} \quad \delta_{u,m}^{\Diamond R} = \varrho_{cp(u)}^{\Diamond R} - \tau_{u,m}^{\Diamond R}.$$

The inductive construction of the domain and sets $S^{n,m}$ will ensure that for each $m \geq 0$, the following holds for all $u \in \Delta_m \backslash \Delta_{m-1}$ (for convenience, we assume that $\Delta_{-1} = \emptyset$):

(fn) $\tau_{u,m}^{\Diamond R} < \omega$, for all $R \in \mathsf{role}_\mathcal{K}$;

(rn) $0 \leq \delta_{u,m}^{\Box R} \leq \delta_{u,m}^{R,n} \leq \delta_{u,m}^{\Diamond R}$, for all $R \in \mathsf{role}_\mathcal{K}$ and all $n \in \mathbb{Z}$;

(df) for all $R \in \mathsf{role}_\mathcal{K}$, if $\delta_{u,m}^{\Box R} < \delta_{u,m}^{\Diamond R}$, then

—$\delta_{u,m}^{\Box R} < \delta_{u,m}^{R,n}$, for infinitely many $n \in \mathbb{Z}$, and
—if additionally $\delta_{u,m}^{\Diamond R} < \omega$, then $\delta_{u,m}^{R,n} < \delta_{u,m}^{\Diamond R}$, for infinitely many $n \in \mathbb{Z}$.

Note that by **(fn)**, $\delta_{u,m}^{\Diamond R}$ and the $\delta_{u,m}^{R,n}$ are well defined and $\delta_{u,m}^{\Diamond R} = \omega$ is just in case $\varrho_{cp(u)}^{\Diamond R} = \omega$.

For the basis of induction ($m = 0$), set $\Delta_0 = \mathsf{ob}_\mathcal{A}$ and $a^\mathcal{I} = a$, for each $a \in \mathsf{ob}_\mathcal{A}$. By **($\mathbf{Q_2}$)**, for each $a \in \Delta_0$, there is a run $r_a \in \mathfrak{R}$ that is a-faithful for $\mathcal{E}$. So, set $cp(a) = r_a$ and take $\nu_a = \nu$, for some fixed function $\nu : \mathbb{Z} \to Z$ such that $\nu(k) = k$ and $\nu^{-1}(k)$ is infinite, for each $k \in Z$. For every role name S, let

$$S^{n,0} = \big\{(a,b) \in \Delta_0 \times \Delta_0 \mid \bigcirc^{\nu(n)} S(a,b) \in \mathcal{E}\big\}, \qquad \text{for } n \in \mathbb{Z}. \tag{68}$$

By definition, $\tau_{a,0}^{\Box R} = |\mathcal{E}_a^{\Box R}|$, $\tau_{a,0}^{R,n} = |\mathcal{E}_a^{R,\nu(n)}|$, for all $n \in \mathbb{Z}$, and $\tau_{a,0}^{\Diamond R} = |\mathcal{E}_a^{\Diamond R}|$. For each $a \in \Delta_0$, **(fn)** is by construction, **(rn)** is immediate from **($\mathbf{r_5}$)**, and **(df)** follows from **($\mathbf{r_6}$)** and the fact that $\nu^{-1}(k)$ is infinite, for each $k \in Z$.

Assuming that Δ_m and the $S^{n,m}$ have been defined and **(fn)**, **(rn)**, and **(df)** hold for some $m \geq 0$, we construct Δ_{m+1} and the $S^{n,m+1}$ and show that the properties also hold for $m+1$. By **(rn)**, for all $u \in \Delta_m$ and $R \in \mathsf{role}_\mathcal{K}$, we have $\delta_{u,m}^{\Box R} \geq 0$, $\delta_{u,m}^{R,n} \geq 0$, for all $n \in \mathbb{Z}$, and $\delta_{u,m}^{\Diamond R} \geq 0$. If these inequalities are actually equalities, then we are done. However, in general, this is not the case, as there may be "defective" elements whose actual rank is smaller than the required rank. Consider the following four sets of defects in $S^{n,m}$, for $R = S$ and $R = S^-$:

$$\Lambda_{\Box R}^m = \big\{u \in \Delta_m \backslash \Delta_{m-1} \mid 0 < \delta_{u,m}^{\Box R}\big\} \quad \text{and} \quad \Lambda_{\Diamond R}^m = \big\{u \in \Delta_m \backslash \Delta_{m-1} \mid \delta_{u,m}^{\Box R} < \delta_{u,m}^{\Diamond R}\big\}.$$

The purpose of $\Lambda_{\Box R}^m$ is to identify elements $u \in \Delta_m \backslash \Delta_{m-1}$ that should have $\varrho_{cp(u)}^{\Box R}$-many distinct $\boxast R$-arrows (according to $\mathfrak{Q}$), but some arrows are still missing (only $\tau_{u,m}^{\Box R}$ arrows exist in Δ_m). The purpose of $\Lambda_{\Diamond R}^m$ is to identify elements u that should have $\varrho_{cp(u)}^{\Diamond R}$-many distinct $\diamondast R$-arrows (according to $\mathfrak{Q}$), but some arrows are still missing—only $\tau_{u,m}^{\Diamond R}$ arrows exist in Δ_m, and $\tau_{u,m}^{\Box R}$ of those are in fact $\boxast R$-arrows. Although $\boxast R$-arrows are also $\diamondast R$-arrows, their defects are repaired using a separate rule; and defects of R-arrows are dealt with as part of repairing defects of $\diamondast R$-arrows. The following rules extend Δ_m to Δ_{m+1} and each $S^{n,m}$ to $S^{n,m+1}$:

($\mathbf{\Lambda_{\Box S}^m}$) If $\delta_{u,m}^{\Box S} > 0$, then $\varrho_{cp(u)}^{\Box S} \geq 1$. By **($\mathbf{Q_3}$)**, there is $r' \in \mathfrak{R}$ such that $\varrho_{r'}^{\Box S^-} \geq 1$. We add $q = \delta_{u,m}^{\Box S}$ copies $v_1, \dots, v_q$ of the run r' to Δ_{m+1} and set $cp(v_i) = r'$, add (u, v_i) to

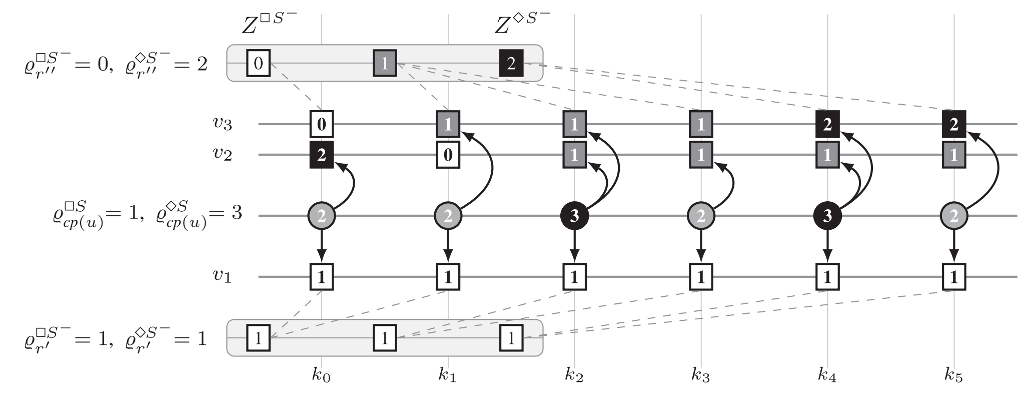
$Z^{\Box S^-}$
$Z^{\Diamond S^-}$
$\varrho^{\Box S^-}_{r''} = 0,\ \varrho^{\Diamond S^-}_{r''} = 2$
v_3
v_2
$\varrho^{\Box S}_{cp(u)} = 1,\ \varrho^{\Diamond S}_{cp(u)} = 3$
v_1
$\varrho^{\Box S^-}_{r'} = 1,\ \varrho^{\Diamond S^-}_{r'} = 1$
k_0
k_1
k_2
k_3
k_4
k_5

Fig. 11. Repairing defects of u: the required S-rank, $\varrho^{S,n}_{cp(u)}$, of u is specified inside the circular nodes. Rule $(\Lambda^m_{\Box S})$ uses run r' to create v_1; $(\Lambda^m_{\Diamond S})$ uses copies of run r'' to create v_2 and v_3. The required S^--rank, $\varrho^{S^-,n}_{cp(v_i)}$, of the created points is specified inside the square nodes. Note that at instant k_3, the element v_3 requires an S-predecessor different from u.

$S^{n,m+1}$, for all $n \in \mathbb{Z}$, and let $\nu_{v_i} : \mathbb{Z} \to Z$ be such that $\nu^{-1}_{v_i}(k)$ is infinite, for each $k \in Z$.

$(\Lambda^m_{\Diamond S})$ Let K be $\{i \mid 0 < i \le \delta^{\Diamond S}_{u,m} - \delta^{\Box S}_{u,m}\}$ if $\delta^{\Diamond S}_{u,m} < \omega$ and $\{i \mid 0 < i \le q_{\mathcal{K}} + 1\}$ otherwise. By assumption, $K \neq \emptyset$. We attach $|K|$ fresh ◈S-successors to u so that the required ⊞S-, S-, and ◈S-ranks coincide with the respective actual ranks at step $m+1$. By **(rn)** and **(df)**, there exists a function $\gamma : \mathbb{Z} \to 2^K$ such that for each $i \in K$, there are infinitely many $n_0 \in \mathbb{Z}$ with $i \notin \gamma(n_0)$ and infinitely many $n_1 \in \mathbb{Z}$ with $i \in \gamma(n_1)$, and for all $n \in \mathbb{Z}$,

$$|\gamma(n)| = \begin{cases} \delta^{S,n}_{u,m} - \delta^{\Box S}_{u,m}, & \text{if } \delta^{S,n}_{u,m} < \omega, \\ q_{\mathcal{K}}, & \text{otherwise.} \end{cases}$$

By assumption, we have $\varrho^{\Box R}_{cp(u)} - \tau^{\Box R}_{u,m} < \varrho^{\Diamond R}_{cp(u)} - \tau^{\Diamond R}_{u,m}$; by definition, $\tau^{\Box R}_{u,m} \le \tau^{\Diamond R}_{u,m}$ and, by **(fn)**, $\tau^{\Diamond R}_{u,m} < \omega$, whence $\varrho^{\Box S}_{cp(u)} < \varrho^{\Diamond S}_{cp(u)}$. Therefore, by **($\mathbf{Q_4}$)**, there exists $r' \in \mathfrak{R}$ with $\varrho^{\Box S^-}_{r'} < \varrho^{\Diamond S^-}_{r'}$. We add $|K|$ *fresh copies* $v_1, \dots, v_{|K|}$ of r' to Δ_{m+1}, and for each $i \in K$, set $cp(v_i) = r'$, and for every $n \in \mathbb{Z}$, add (u, v_i) to $S^{n,m+1}$ iff $i \in \gamma(n)$. Let

$$Z^{\Box S^-} = \{k \in Z \mid \varrho^{\Box S^-}_{r'} = \varrho^{S^-,k}_{r'}\}, \quad Z^{\Diamond S^-} = \begin{cases} \{k \in Z \mid \varrho^{S^-,k}_{r'} = \varrho^{\Diamond S^-}_{r'}\}, & \text{if } \varrho^{\Diamond S^-}_{r'} < \omega, \\ \emptyset, & \text{otherwise.} \end{cases}$$

For each v_i, we take a function $\nu_{v_i} : \mathbb{Z} \to Z$ such that each $\nu^{-1}_{v_i}(k)$ is infinite, for $k \in Z$, and

—if $k \in Z^{\Box S^-}$, then $i \notin \gamma(n)$, for each $n \in \nu^{-1}_{v_i}(k)$;

—if $k \in Z \setminus (Z^{\Box S^-} \cup Z^{\Diamond S^-})$, then $i \in \gamma(n)$ for infinitely many $n \in \nu^{-1}_{v_i}(k)$ and $i \notin \gamma(n)$ for infinitely many $n \in \nu^{-1}_{v_i}(k)$;

—if $k \in Z^{\Diamond S^-}$, then $i \in \gamma(n)$, for each $n \in \nu^{-1}_{v_i}(k)$

(Figure 11). Intuitively, if k is such that not every S-predecessor is required to be a ⊞S-predecessor, then there should be infinitely many copies of k with $(u, v_i) \in S^{n,m+1}$; symmetrically, if k is such that not every ◈ S-predecessor is required to be an S-predecessor, then there should be infinitely many copies of k with $(u, v_i) \notin S^{n,m+1}$.

$(\Lambda^m_{\Box S^-})$ and **$(\Lambda^m_{\Diamond S^-})$** are the mirror images of **$(\Lambda^m_{\Box S})$** and **$(\Lambda^m_{\Diamond S})$**, respectively.

By construction, the rules guarantee that for any $m \geq 0$ and $u \in \Delta_m$,

$$0 = \delta^{\Box R}_{u,m+1} = \delta^{R,n}_{u,m+1} = \delta^{\Diamond R}_{u,m+1}, \text{ for all } R \in \mathsf{role}_{\mathcal{K}} \text{ and all } n \in \mathbb{Z}. \tag{69}$$

We now show that **(fn)**, **(rn)**, and **(df)** hold for each $v \in \Delta_{m+1} \setminus \Delta_m$. Indeed, **(fn)** holds because $\tau^{\Diamond R}_{v,m+1} \leq 1$. In the case of $(\boldsymbol{\Lambda}^{\boldsymbol{m}}_{\Box \mathbf{S}})$, property **(rn)** follows from

$$1 = \tau^{\Box S^-}_{v,m+1} = \tau^{S^-,\, n}_{v,m+1} = \tau^{\Diamond S^-}_{v,m+1} \leq \varrho^{\Box S^-}_{cp(v)} \leq \varrho^{S^-,\, \nu_v(n)}_{cp(v)} \leq \varrho^{\Diamond S^-}_{cp(v)}, \tag{70}$$

$$0 = \tau^{\Box R}_{v,m+1} = \tau^{R,n}_{v,m+1} = \tau^{\Diamond R}_{v,m+1} \leq \varrho^{\Box R}_{cp(v)} \leq \varrho^{R,\, \nu_v(n)}_{cp(v)} \leq \varrho^{\Diamond R}_{cp(v)}, \quad \text{for all } R \neq S^-. \tag{71}$$

Then, **(df)** is by the definition of ν_v. The case of $(\boldsymbol{\Lambda}^{\boldsymbol{m}}_{\Box \mathbf{S}^-})$ is similar. For the case of $(\boldsymbol{\Lambda}^{\boldsymbol{m}}_{\Diamond \mathbf{S}})$, we observe that for each $R \neq S^-$, we have (71), and so **(rn)** and **(df)** follow as earlier. Let us consider S^-. By $(\mathbf{r_2})$, both $Z \setminus Z^{\Diamond S^-}$ and $Z \setminus Z^{\Box S^-}$ are nonempty. It follows that $\tau^{\Box S^-}_{v,m+1} = 0$ and $\tau^{\Diamond S^-}_{v,m+1} = 1$. By definition, we also have **(rn)**. To show **(df)**, suppose that $\varrho^{\Box S^-}_{cp(v)} < \varrho^{\Diamond S^-}_{cp(v)} - 1$. Clearly, $Z^{\Box S^-} \cap Z^{\Diamond S^-} = \emptyset$. If there is $k \in Z^{\Diamond S^-}$, then $\varrho^{S^-,\, k}_{cp(v)} = \varrho^{\Diamond S^-}_{cp(v)}$ and $\tau^{S^-,\, n}_{v,m+1} = 1$, for all (infinitely many) $n \in \nu_v^{-1}(k)$, whence the first item of **(df)** holds; otherwise, there is $k \in Z \setminus (Z^{\Box S^-} \cup Z^{\Diamond S^-})$ and therefore $\varrho^{\Box S^-}_{cp(v)} < \varrho^{S^-,\, k}_{cp(v)}$ and $\tau^{S^-,\, n}_{v,m+1} = 0$, for infinitely many $n \in \nu_v^{-1}(k)$, whence the first item of **(df)** holds. The second item of **(df)** is obtained by a symmetric argument.

The definition of $\mathcal{I}$ is completed by taking $A^{\mathcal{I}(n)} = \{u \in \Delta^{\mathcal{I}} \mid A \in r(\nu_u(n)),\ r = cp(u)\}$, for each concept name A. Observe that $\mathcal{I} \models \mathcal{E}$ because each ν_a, $a \in \mathsf{ob}_{\mathcal{A}}$, coincides with the fixed ν. Next, we show by induction on the construction of concepts C in $\mathcal{K}$ that

$$C \in r(\nu_u(n)) \text{ with } r = cp(u) \qquad \text{iff} \qquad u \in C^{\mathcal{I}(n)}, \qquad \text{for all } n \in \mathbb{Z} \text{ and } u \in \Delta^{\mathcal{I}}.$$

The basis of induction is by definition for $C = \bot$ and $C = A_i$; for $C = \geq q\, R$, it follows from (69) and the fact that arrows to $u \in \Delta_m \setminus \Delta_{m-1}$ can be added only at steps m and $m+1$ as part of the defect repair process. The induction step for $C = \neg C_1$ and $C = C_1 \sqcap C_2$ follows from the induction hypothesis by $(\mathbf{t_1})$ and $(\mathbf{t_2})$, respectively. The induction step for $C = \circledast\, C_1$ follows from the induction hypothesis by $(\mathbf{t_1})$, $(\mathbf{t_3})$, $(\mathbf{r_1})$, and $(\mathbf{r_3})$. Thus, by $(\mathbf{Q_1})$, $\mathcal{I} \models \mathcal{T}$.

It remains to show $\mathcal{I} \models \mathcal{A}$. By the definition of $\mathcal{E}$ and $\mathcal{I}$, if $\bigcirc^k A(a) \in \mathcal{A}$, then $\mathcal{I} \models \bigcirc^k A(a)$, and if $\bigcirc^k \neg A(a) \in \mathcal{A}$, then $\mathcal{I} \models \bigcirc^k \neg A(a)$. If $\bigcirc^k S(a,b) \in \mathcal{A}$, then, by (68), $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in S^{k,0}$, whence $\mathcal{I} \models \bigcirc^k S(a,b)$. If $\bigcirc^k \neg S(a,b) \in \mathcal{A}$, then $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin \mathcal{E}$, whence $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin S^{k,0}$ by (68), and so $\mathcal{I} \models \bigcirc^k \neg S(a,b)$ as no new arrows can be added between ABox individuals. $\Box$

We are now in a position to establish the NP membership of the satisfiability problem for $T^*_U DL\text{-}Lite^{\mathcal{N}}_{bool}$ KBs. To check whether a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is satisfiable, it is enough to guess a structure $\mathfrak{Q} = (Q, Z, \mathfrak{R}, \mathcal{E})$ consisting of a set $\mathfrak{R}$ of runs and an extension $\mathcal{E}$ of the ABox $\mathcal{A}$, both of which are of polynomial size in $|\mathcal{K}|$, and check that $\mathfrak{Q}$ is a quasimodel for $\mathcal{K}$. NP-hardness follows from the complexity of $DL\text{-}Lite_{bool}$. This completes the proof of Theorem 6.3.

7. CONCLUSIONS

Logics interpreted over 2D (or more) Cartesian products are notorious for their bad computational properties, which is well documented in the modal logic literature (e.g., see Gabbay et al. [2003] and Kurucz [2007] and references therein). For example, satisfiability of bimodal formulas over Cartesian products of transitive Kripke frames is undecidable [Gabelaia et al. 2005]; by dropping the requirement of transitivity, we gain decidability but the complexity is not elementary [Göller et al. 2012]; if one dimension is a linear-time line, then the complexity can only become worse [Gabbay et al. 2003].

The principal achievement of this article is the construction of TDLs that (1) are interpreted over 2D Cartesian products, (2) are capable of capturing standard temporal conceptual modelling constraints, and (3) in many cases are of reasonable computational complexity. Although TDLs $T^*_{FP}\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and $T^*_{X}\textit{DL-Lite}^{\mathcal{N}}_{bool}$, capturing lifespan cardinalities together with qualitative or quantitative evolution, turned out to be undecidable (as well as TDLs with unrestricted role inclusions), the complexity of the remaining 10 logics ranges between NLogSpace and PSpace. We established these positive results by reductions to various clausal fragments of propositional temporal logic (the complexity analysis of which could be of interest on its own). We have conducted initial experiments, using two off-the-shelf temporal reasoning tools, NuSmv [Cimatti et al. 2002] and TeMP [Hustadt et al. 2004], which showed feasibility of automated reasoning over TCMs with both timestamping and evolution constraints but without subrelations ($T_{FPX}\textit{DL-Lite}^{\mathcal{N}}_{bool}$). Many efficiency issues are yet to be resolved, but the first results are encouraging.

The most interesting TDLs not considered in this article are probably $T^*_{FPX}\textit{DL-Lite}^{\mathcal{N}}_{core}$ and $T^*_{FPX}\textit{DL-Lite}^{\mathcal{N}}_{krom}$. We conjecture that both of them are decidable. We also believe that the former can be used as a variant of temporal RDFS (cf. Gutiérrez et al. [2005]).

Although the results in this article establish tight complexity bounds for TDLs, they can only be used to obtain upper complexity bounds for the corresponding fragments of TCMs; the lower bounds are mostly left for future work [Artale et al. 2010].

The original *DL-Lite* family [Calvanese et al. 2007] was designed with the primary aim of OBDA by means of first-order query rewriting. In fact, OBDA has already reached a mature stage and has become a prominent direction in the development of the next generation of information systems and the Semantic Web (e.g., see Polleres et al. [2013] and Kontchakov et al. [2013] for recent surveys and references therein). In particular, W3C has introduced a special profile, OWL 2 QL, of the Web Ontology Language OWL 2 that is suitable for OBDA and based on the *DL-Lite* family. An interesting problem, both theoretically and practically, is to investigate how far this approach can be developed in the temporal case and what temporal ontology languages can support first-order query rewriting (e.g., see Gutiérrez-Basulto and Klarman [2012], Motik [2012], Artale et al. [2013], Baader et al. [2013], and Borgwardt et al. [2013] for some initial results).

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

REFERENCES

Tom Apostol. 1976. *Introduction to Analytic Number Theory*. Springer.

Alessandro Artale, Diego Calvanese, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. 2007a. Reasoning over extended ER models. In *Proceedings of the 26th International Conference on Conceptual Modeling (ER'07)*. Lecture Notes in Computer Science, Vol. 4801. Springer-Verlag, Berlin, Heidelberg, 277–292.

Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. 2007b. *DL-Lite* in the light of first-order logic. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI'07)*. 361–366.

Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. 2009a. The *DL-Lite* family and relations. *Journal of Artificial Intelligence Research* 36, 1–69.

Alessandro Artale and Enrico Franconi. 1998. A temporal description logic for reasoning about actions and plans. *Journal of Artificial Intelligence Research* 9, 463–506.

Alessandro Artale and Enrico Franconi. 1999. Temporal ER modeling with description logics. In *Proceedings of the 18th International Conference on Conceptual Modeling (ER'99)*. Lecture Notes in Computer Science, Vol. 1728. Springer-Verlag, Berlin, Heidelberg, 81–95.

Alessandro Artale and Enrico Franconi. 2001. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence* 30, 1–4, 171–210.

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

Alessandro Artale and Enrico Franconi. 2009. Foundations of temporal conceptual data models. In *Conceptual Modeling: Foundations and Applications*. Lecture Notes in Computer Science, Vol. 5600. Springer-Verlag, Berlin, Heidelberg, 10–35.

Alessandro Artale, Enrico Franconi, and Federica Mandreoli. 2003. Description logics for modelling dynamic information. In *Logics for Emerging Applications of Databases*. Springer, 239–275.

Alessandro Artale, Enrico Franconi, Frank Wolter, and Michael Zakharyaschev. 2002. A temporal description logic for reasoning about conceptual schemas and queries. In *Proceedings of the 8th Joint European Conference on Logics in Artificial Intelligence (JELIA'02).* Lecture Notes in Artificial Intelligence, Vol. 2424. Springer-Verlag, Berlin, Heidelberg, 98–110.

Alessandro Artale, Roman Kontchakov, Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. 2007c. Temporalising tractable description logics. In *Proceedings of the 14th International Symposium on Temporal Representation and Reasoning (TIME'07)*. IEEE Computer Society, Washington, DC, 11–22.

Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. 2009b. *DL-Lite* with temporalised concepts, rigid axioms and roles. In *Proceedings of the 7th International Symposium on Frontiers of Combining Systems (FroCoS'09).* Lecture Notes in Computer Science, Vol. 5749. Springer-Verlag, Berlin, Heidelberg, 133–148.

Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. 2010. Complexity of reasoning over temporal data models. In *Proceedings of the 29th International Conference on Conceptual Modeling (ER'10).* Lecture Notes in Computer Science, Vol. 4801. Springer-Verlag, Berlin, Heidelberg, 277–292.

Alessandro Artale, Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. 2013. Temporal description logic for ontology-based data access. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*. 711–717.

Alessandro Artale, Carsten Lutz, and David Toman. 2007d. A description logic of change. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*. 218–223.

Alessandro Artale, Christine Parent, and Stefano Spaccapietra. 2007e. Evolving objects in temporal information systems. *Annals of Mathematics and Artificial Intelligence* 50, 1–2, 5–38.

Franz Baader, Stefan Borgwardt, and Marcel Lippmann. 2013. Temporalizing ontology-based data access. In *Proceedings of the 24th International Conference on Automated Deduction (CADE'13).* Lecture Notes in Computer Science, Vol. 7898. Springer-Verlag, Berlin, Heidelberg, 330–344.

Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider (Eds.). 2003. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

Franz Baader, Silvio Ghilardi, and Carsten Lutz. 2008. LTL over description logic axioms. In *Proceedings of the 11th International Conference on the Principles of Knowledge Representation and Reasoning (KR'08)*. 684–694.

Franz Baader, Silvio Ghilardi, and Carsten Lutz. 2012. LTL over description logic axioms. *ACM Transactions on Computational Logic* 13, 3, 21.

Michael Bauland, Thomas Schneider, Henning Schnoor, Ilka Schnoor, and Heribert Vollmer. 2009. The complexity of generalized satisfiability for linear temporal logic. *Logical Methods in Computer Science* 5, 1, 48–62.

Daniela Berardi, Diego Calvanese, and Guiseppe De Giacomo. 2005. Reasoning on UML class diagrams. *Artificial Intelligence* 168, 1–2, 70–118.

Claudio Bettini. 1997. Time dependent concepts: Representation and reasoning using temporal description logics. *Data and Knowledge Engineering* 22, 1, 1–38.

Egon Börger, Erich Grädel, and Yuri Gurevich. 1997. *The Classical Decision Problem*. Springer.

Alexander Borgida and Ronald J. Brachman. 2003. Conceptual modeling with description logics. In *The Description Logic Handbook*, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider (Eds.). Cambridge University Press, 349–372.

Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. 2013. Temporal query answering in the description logic *DL-Lite.* In *Proceedings of the 9th International Symposium on Frontiers of Combining Systems (FroCoS'13).* Lecture Notes in Computer Science, Vol. 8152. Springer-Verlag, Berlin, Heidelberg, 165–180.

Diego Calvanese, Guiseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. 2005. *DL-Lite*: Tractable description logics for ontologies. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI'05)*. 602–607.

Diego Calvanese, Guiseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. 2007. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated Reasoning* 39, 3, 385–429.

Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Daniele Nardi. 2001. Reasoning in expressive description logics. In *Handbook of Automated Reasoning*. Vol. II. Elsevier Science Publishers, 1581–1634.

Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. 1999. Unifying class-based representation formalisms. *Journal of Artificial Intelligence Research* 11, 199–240.

Cheng-Chia Chen and I.-Peng Lin. 1993. The computational complexity of satisfiability of temporal Horn formulas in propositional linear-time temporal logic. *Information Processing Letters* 45, 3, 131–136.

Peter Pin-Shan Chen. 1976. The entity-relationship model—toward a unified view of data. *ACM Transactions on Database Systems* 1, 9–36.

Jan Chomicki, David Toman, and Michael H. Böhlen. 2001. Querying ATSQL databases with temporal logic. *ACM Transactions on Database Systems* 26, 2, 145–178.

Marek Chrobak. 1986. Finite automata and unary languages. *Theoretical Computer Science* 47, 2, 149–158.

Alessandro Cimatti, Edmund M. Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella. 2002. NuSMV 2: An opensource tool for symbolic model checking. In *Proceedings of the 14th International Conference on Computer Aided Verification (CAV'02).* Lecture Notes in Computer Science, Vol. 2404. Springer-Verlag, Berlin, Heidelberg, 359–364.

Carlo Combi, Sara Degani, and Christian S. Jensen. 2008. Capturing temporal constraints in temporal ER models. In *Proceedings of the 27th International Conference on Conceptual Modeling (ER'08).* Lecture Notes in Computer Science, Vol. 5231. Springer-Verlag, Berlin, Heidelberg, 397–411.

Anatoli Degtyarev, Michael Fisher, and Boris Konev. 2006. Monodic temporal resolution. *ACM Transactions on Computational Logic* 7, 1, 108–150.

Stephane Demri and Phillipe Schnoebelen. 2002. The complexity of propositional linear temporal logics in simple cases. *Information and Computation* 174, 1, 84–103.

Clare Dixon, Michael Fisher, and Boris Konev. 2007. Tractable temporal reasoning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*. 318–323.

Julian Dolby, Achille Fokoue, Aditya Kalyanpur, Li Ma, Edith Schonberg, Kavitha Srinivas, and Xingzhi Sun. 2008. Scalable grounded conjunctive query evaluation over large and expressive knowledge bases. In *Proceedings of the 7th International Semantic Web Conference (ISWC'08).* Lecture Notes in Computer Science, Vol. 5318. Springer-Verlag, Berlin, Heidelberg, 403–418.

Ramez A. Elmasri and Shamkant B. Navathe. 2007. *Fundamentals of Database Systems* (5th ed.). Addison Wesley.

Marcelo Finger and Peter McBrien. 2000. Temporal conceptual-level databases. In *Temporal Logic: Mathematical Foundations and Computational Aspects*, D. M. Gabbay, M. A. Reynolds, and M. Finger (Eds.). Oxford University Press, 409–435.

Michael Fisher. 1991. A resolution method for temporal logic. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI'91)*. Morgan Kaufmann, 99–104.

Michael Fisher, Clare Dixon, and Martin Peim. 2001. Clausal temporal resolution. *ACM Transactions on Computational Logic* 2, 1, 12–56.

Dov Gabbay, Marcelo Finger, and Mark Reynolds. 2000. *Temporal Logic: Mathematical Foundations and Computational Aspects*. Vol. 2. Oxford University Press.

Dov Gabbay, Ian Hodkinson, and Mark Reynolds. 1994. *Temporal Logic: Mathematical Foundations and Computational Aspects*. Vol. 1. Oxford University Press.

Dov Gabbay, Agi Kurucz, Frank Wolter, and Michael Zakharyaschev. 2003. *Many-Dimensional Modal Logics: Theory and Applications*. Studies in Logic and the Foundations of Mathematics, Vol. 148. Elsevier.

David Gabelaia, Agi Kurucz, Frank Wolter, and Michael Zakharyaschev. 2005. Products of "transitive" modal logics. *Journal of Symbolic Logic* 70, 3, 993–1021.

Stefan Göller, Jean Christoph Jung, and Markus Lohrey. 2012. The complexity of decomposing modal and first-order theories. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science (LICS'12)*. IEEE Computer Society, Washington, DC, 325–334.

Heidi Gregersen and Christian S. Jensen. 1998. *Conceptual Modeling of Time-Varying Information*. Technical Report TimeCenter TR-35. Aalborg University, Denmark.

Heidi Gregersen and Christian S. Jensen. 1999. Temporal entity-relationship models: A survey. *IEEE Transactions on Knowledge and Data Engineering* 11, 3, 464–497.

Christian Guensel. 2005. *A Tableaux-Based Reasoner for Temporalised Description Logics*. Ph.D. Dissertation. University of Liverpool.

Claudio Gutiérrez, Carlos A. Hurtado, and Alejandro A. Vaisman. 2005. Temporal RDF. In *Proceedings of the 2nd European Semantic Web Conference (ESWC'05)*. Lecture Notes in Computer Science, Vol. 3532. Springer-Verlag, Berlin, Heidelberg, 93–107.

Víctor Gutiérrez-Basulto and Szymon Klarman. 2012. Towards a unifying approach to representing and querying temporal data in description logics. In *Proceedings of the 6th International Conference on Web Reasoning and Rule Systems (RR'12)*. Lecture Notes in Computer Science, Vol. 7497. Springer-Verlag, Berlin, Heidelberg, 90–105.

Gary Hall and Ranabir Gupta. 1991. Modeling transition. In *Proceedings of the 7th International Conference on Data Engineering (ICDE'91)*. IEEE Computer Society, Washington, DC, 540–549.

Joseph Y. Halpern and John H. Reif. 1981. The propositional dynamic logic of deterministic, well-structured programs (extended abstract). In *Proceedings of the 22nd Annual Symposium on Foundations of Computer Science (FOCS'81)*. IEEE Computer Society, Washington, DC, 322–334.

Joseph Y. Halpern and Yoav Shoham. 1991. A propositional modal logic of time intervals. *Journal of the ACM* 38, 4, 935–962.

Stijn Heymans, Li Ma, Darko Anicic, Zhilei Ma, Nathalie Steinmetz, Yue Pan, Jing Mei, Achille Fokoue, Aditya Kalyanpur, Aaron Kershenbaum, Edith Schonberg, Kavitha Srinivas, Cristina Feier, Graham Hench, Branimir Wetzstein, and Uwe Keller. 2008. Ontology reasoning with large data repositories. In *Ontology Management, Semantic Web, Semantic Web Services, and Business Applications*. Vol. 7. Springer, 89–128.

Ian Hodkinson, Frank Wolter, and Michael Zakharyaschev. 2000. Decidable fragments of first-order temporal logics. *Annals of Pure and Applied Logic* 106, 85–134.

Ullrich Hustadt, Boris Konev, Alexandre Riazanov, and Andrei Voronkov. 2004. TeMP: A temporal monodic prover. In *Proceedings of the 2nd International Joint Conference on Automated Reasoning (IJCAR'04)*. Lecture Notes in Computer Science, Vol. 3097. Springer-Verlag, Berlin, Heidelberg, 326–330.

Christian S. Jensen and Richard T. Snodgrass. 1999. Temporal data management. *IEEE Transactions on Knowledge and Data Engineering* 111, 1, 36–44.

Roman Kontchakov, Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. 2004. Temporalising tableaux. *Studia Logica* 76, 1, 91–134.

Roman Kontchakov, Mariano Rodríguez-Muro, and Michael Zakharyaschev. 2013. Ontology-based data access with databases: A short course. In *Reasoning Web. The 9th International Summer School on Semantic Technologies for Intelligent Data Access.* Lecture Notes in Computer Science, Vol. 8067. Springer-Verlag, Berlin, Heidelberg, 194–229.

Agi Kurucz. 2007. Combining modal logics. In *Handbook of Modal Logic*, P. Blackburn, J. van Benthem, and F. Wolter (Eds.). Studies in Logic and Practical Reasoning, Vol. 3. Elsevier, 869–924.

Orna Lichtenstein, Amir Pnueli, and Lenore D. Zuck. 1985. The glory of the past. In *Proceedings of the Conference on Logic of Programs.* Lecture Notes in Computer Science, Vol. 193. Springer-Verlag, Berlin, Heidelberg, 196–218.

Michel Ludwig and Ullrich Hustadt. 2010. Implementing a fair monodic temporal logic prover. *AI Communications* 23, 2–3, 69–96.

Carsten Lutz, Holger Sturm, Frank Wolter, and Michael Zakharyaschev. 2002. A tableau decision algorithm for modalized ALC with constant domains. *Studia Logica* 72, 2, 199–232.

Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. 2008. Temporal description logics: A survey. In *Proceedings of the 15th International Symposium on Temporal Representation and Reasoning (TIME'08)*. IEEE Computer Society, Washington, DC, 3–14.

Nicolas Markey. 2004. Past is for free: On the complexity of verifying linear temporal properties with past. *Acta Informatica* 40, 6–7, 431–458.

Peter McBrien, Anna H. Seltveit, and Benkt Wangler. 1992. An entity-relationship model extended to describe historical information. In *Proceedings of the International Conference on Information Systems and Management of Data (CISMOD'92)*. 244–260.

Alberto O. Mendelzon, Tova Milo, and Emmanuel Waller. 1994. Object migration. In *Proceedings of the 13th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'94)*. ACM Press, New York, NY, 232–242. DOI:http://dx.doi.org/10.1145/182591.182616

Boris Motik. 2012. Representing and querying validity time in RDF and OWL: A logic-based approach. *Journal of Web Semantics* 12, 3–21.

Hiroakira Ono and Akira Nakamura. 1980. On the size of refutation Kripke models for some linear modal and tense logics. *Studia Logica* 39, 325–333.

Christos M. Papadimitriou. 1994. *Computational Complexity*. Addison-Wesley, Reading, MA.

Christine Parent, Stefano Spaccapietra, and Esteban Zimanyi. 2006. *Conceptual Modeling for Traditional and Spatio-Temporal Applications: The MADS Approach*. Springer.

David Plaisted. 1986. A decision procedure for combinations of propositional temporal logic and other specialized theories. *Journal of Automated Reasoning* 2, 171–190.

Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. 2008. Linking data to ontologies. *Journal on Data Semantics* X, 133–173.

Axel Polleres, Aidan Hogan, Renaud Delbru, and Jurgen Umbrich. 2013. RDFS and OWL reasoning for linked data. In *Reasoning Web. The 9th International Summer School on Semantic Technologies for Intelligent Data Access.* Lecture Notes in Computer Science, Vol. 8067. Springer-Verlag, Berlin, Heidelberg, 91–149.

Alexander Rabinovich. 2010. Temporal logics over linear time domains are in PSPACE. In *Proceedings of the 4th International Workshop on Reachability Problems.* Lecture Notes in Computer Science, Vol. 6227. Springer-Verlag, Berlin, Heidelberg, 29–50.

Mark Reynolds. 2010. The complexity of decision problems for linear temporal logics. *Journal of Studies in Logic* 3, 1, 19–50.

Klaus Schild. 1993. Combining terminological logics with tense logic. In *Proceedings of the 6th Portuguese Conference on Artificial Intelligence (EPIA'93)*. Springer-Verlag, Berlin, Heidelberg, 105–120.

Albrecht Schmiedel. 1990. A temporal terminological logic. In *Proceedings of the 8th National Conference on Artificial Intelligence (AAAI'90)*. 640–645.

A. Prasad Sistla and Edmund M. Clarke. 1982. The complexity of propositional linear temporal logics. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing (STOC'82)*. ACM Press, New York, NY, 159–168.

Larry J. Stockmeyer and Albert R. Meyer. 1973. Word problems requiring exponential time: Preliminary report. In *Proceedings of the 5th Annual ACM Symposium on Theory of Computing (STOC'73)*. ACM Press, New York, NY, 1–9.

Jianwen Su. 1997. Dynamic constraints and object migration. *Theoretical Computer Science* 184, 1–2, 195–236.

Branka Tauzovich. 1991. Towards temporal extensions to the entity-relationship model. In *Proceedings of the 10th International Conference on Conceptual Modeling (ER'91)*. 163–179.

Charalampos I. Theodoulidis, Pericles Loucopoulos, and Benkt Wangler. 1991. A conceptual modelling formalism for temporal database applications. *Information Systems* 16, 3, 401–416.

Anthony Widjaja To. 2009. Unary finite automata vs. arithmetic progressions. *Information Processing Letters* 109, 17, 1010–1014. DOI:http://dx.doi.org/10.1016/j.ipl.2009.06.005

Frank Wolter and Michael Zakharyaschev. 1999. Modal description logics: Modalizing roles. *Fundamenta Informaticæ* 39, 411–438.

Received September 2012; revised October 2013; accepted February 2014