A Temporal Description Logic for Reasoning over Conceptual Schemas and Queries

Alessandro Artale[1], Enrico Franconi[2], Frank Wolter[3], Michael Zakharyaschev[4]

[1] Dept. of Computation, UMIST, Manchester, UK; artale@co.umist.ac.uk
[2] Faculty of Computer Science, Free Univ. of Bolzano, Italy; franconi@inf.unibz.it
[3] Inst. für Informatik, Univ. of Leipzig, D; wolter@informatik.uni-leipzig.de
[4] Dept. of Computer Science, Kings College, London, UK; mz@dcs.kcl.ac.uk

Abstract. This paper introduces a new logical formalism, intended for temporal conceptual modelling, as a natural combination of the well-known description logic $\mathcal{DLR}$ and point-based linear temporal logic with *Since* and *Until*. We define a query language (where queries are non-recursive Datalog programs and atoms are complex $\mathcal{DLR}_{\mathcal{US}}$ expressions) and investigate the problem of checking query containment under the constraints defined by $\mathcal{DLR}_{\mathcal{US}}$ conceptual schemas—i.e., $\mathcal{DLR}_{\mathcal{US}}$ knowledge bases—as well as the problems of schema satisfiability and logical implication.

1 Introduction

Temporal information systems are information systems that store historical information, i.e., past, present, and potential future data. Many formalisations have been proposed for temporal information systems which are based on first-order temporal logic [14]. Although these formalisations can be very useful for characterising semantical problems arising in temporalised ontologies and in temporal databases, like conceptual modelling or querying, usually they are computationally unfeasible for performing deduction tasks (for example, logical implication in the first-order temporal logic of the flow of time $\langle \mathbb{Z}, < \rangle$ or $\langle \mathbb{N}, < \rangle$ is not even recursively enumerable). Note that we are interested in deduction rather than model checking. An obvious solution to this problem would be to look for well-behaved fragments of first-order temporal logic (see e.g. [14] and references therein); however this way has not been successful—the only promising approach we know of is the recent paper [19]. Another idea is to deviate from the first-order paradigm and start from computationally more friendly languages such as description logics which have been used in the area of non-temporal information management to characterise in a uniform framework both conceptual modelling and queries [9, 6, 7].

The *temporal description logic* $\mathcal{DLR}_{\mathcal{US}}$ we devise in this paper is based on the expressive and decidable description logic $\mathcal{DLR}$ which allows the logical reconstruction and the extension of representational tools such as object-oriented and semantic data models, frame-based and web ontology languages [10, 11]. In this setting, an interesting feature of $\mathcal{DLR}$ is the ability to completely *define* classes and relations as $\mathcal{DLR}$ views over other classes and relations of the conceptual schema. Moreover, $\mathcal{DLR}$ formulas can express a large class of integrity constraints that are typical in databases, for instance, existence dependencies, exclusion dependencies, typed inclusion dependencies without projection of relations, unary inclusion dependencies, full key dependencies [9]. Logical implication in $\mathcal{DLR}$ is EXPTIME-complete [9]; practical correct and complete algorithms exist, used in conceptual modelling applications [20, 16].

$\mathcal{DLR}$ is not only a very powerful language for conceptual modelling. The problem of view-based query processing under $\mathcal{DLR}$ constraints has also been studied [9]. View-based query answering requires to answer a query over a virtual database (constrained by a $\mathcal{DLR}$ theory playing the role of the conceptual schema and of the integrity constraints) for which the only information comes from a set of materialised views over the same database; this problem with non-recursive Datalog queries and views is a co-NP-complete problem (in data complexity) under the closed world assumption. Checking query containment of non-recursive Datalog queries under $\mathcal{DLR}$ constraints is decidable in 2EXPTIME [9].

Given all these nice features of $\mathcal{DLR}$, it is natural to try to extend it with a temporal dimension, to understand the expressive power of the resulting hybrid with respect to the needs of temporal conceptual modelling and view based query processing, and to investigate its computational properties. This paper reports the results of such an attempt. We construct $\mathcal{DLR}_{\mathcal{US}}$ as an organic combination of $\mathcal{DLR}$ and the propositional linear temporal logic with *Since* and *Until* (which usually serves as the temporal component in the first-order approach) by allowing applications of temporal operators to all syntactical terms of $\mathcal{DLR}$: classes, relations, and formulas. We then investigate computational properties of reasoning with $\mathcal{DLR}_{\mathcal{US}}$ by analysing schema, class, and relation satisfiability, logical implication, and query containment for non-recursive Datalog queries under $\mathcal{DLR}_{\mathcal{US}}$ constraints.

The full $\mathcal{DLR}_{\mathcal{US}}$ turns out to be undecidable. The main reason for this is the possibility to postulate that a binary relation does not vary in time—a very small fragment of $\mathcal{DLR}_{\mathcal{US}}$ (say, $\mathcal{DLR}$ augmented with a single time invariant binary relation) can encode the undecidable tiling problem (cf. [26, 19]). The fragment $\mathcal{DLR}^{-}_{\mathcal{US}}$ of $\mathcal{DLR}_{\mathcal{US}}$ deprived of the ability to talk about temporal persistence of n-ary relations, for $n \geq 2$, is still very expressive, as is illustrated by examples in this paper, but its computational behaviour is much better. We obtain the following non-trivial novel complexity results: (1) reasoning in $\mathcal{DLR}^{-}_{\mathcal{US}}$ with atomic formulas is EXPTIME-complete, (2) satisfiability and logical implication of arbitrary $\mathcal{DLR}^{-}_{\mathcal{US}}$ formulas is EXPSPACE-complete, and (3) the problem of checking query containment of non-recursive Datalog queries under $\mathcal{DLR}^{-}_{\mathcal{US}}$ constraints is decidable in 2EXPTIME with an EXPSPACE lower bound.

The results obtained in this paper are novel for several reasons. Previous approaches to temporal description logics considered much weaker languages having only binary relations (i.e., roles), without the cardinality constructs, without the inverse construct (which $\mathcal{DLR}_{\mathcal{US}}$ implicitly is able to express by considering only binary relations), and without ever considering the ability to express queries [21, 25, 28, 23]. In this paper for the first time an upper bound for the complexity of reasoning in a temporal description logic with both future and past operators is proved, leading to a tight EXPSPACE-completeness result which automatically holds for the weaker basic temporal description logic $\mathcal{ALC}^{-}_{\mathcal{US}}$ as well. In this paper for the first time non trivial decidability and complexity results are presented for the problem of temporal query containment under complex constraints. For a survey of the previous various approaches to temporal description logics see [4].

2 The temporal description logic

In this paper, we adopt the classical *snapshot* representation of abstract temporal databases (see e.g. [14]). The flow of time $\mathcal{T} = \langle \mathcal{T}_p, < \rangle$, where $\mathcal{T}_p$ is a set of time points (or chronons) and $<$ a binary precedence relation on $\mathcal{T}_p$, is assumed to be isomorphic to $\langle \mathbb{Z}, < \rangle$. Thus, a temporal database can be regarded as a mapping from time points in $\mathcal{T}$

$$
\begin{aligned}
R \rightarrow\ & \top_n \mid RN \mid \neg R \mid R_1 \sqcap R_2 \mid R_1 \sqcup R_2 \mid U_i/n : E \mid \\
& \Diamond^+ R \mid \Diamond^- R \mid \Box^+ R \mid \Box^- R \mid \oplus R \mid \ominus R \mid R_1\, \mathcal{U}\, R_2 \mid R_1\, \mathcal{S}\, R_2 \\
E \rightarrow\ & \top \mid EN \mid \neg E \mid E_1 \sqcap E_2 \mid E_1 \sqcup E_2 \mid \exists^{\lesseqgtr k}[U_j]R \mid \\
& \Diamond^+ E \mid \Diamond^- E \mid \Box^+ E \mid \Box^- E \mid \oplus E \mid \ominus E \mid E_1\, \mathcal{U}\, E_2 \mid E_1\, \mathcal{S}\, E_2
\end{aligned}
$$

$$
\begin{aligned}
(\top_n)^{\mathcal{I}(t)} &\subseteq (\Delta^{\mathcal{I}})^n \\
RN^{\mathcal{I}(t)} &\subseteq (\top_n)^{\mathcal{I}(t)} \\
(\neg R)^{\mathcal{I}(t)} &= (\top_n)^{\mathcal{I}(t)} \setminus R^{\mathcal{I}(t)} \\
(R_1 \sqcap R_2)^{\mathcal{I}(t)} &= R_1^{\mathcal{I}(t)} \cap R_2^{\mathcal{I}(t)} \\
(U_i/n : E)^{\mathcal{I}(t)} &= \{ \langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid d_i \in E^{\mathcal{I}(t)} \} \\
(R_1\, \mathcal{U}\, R_2)^{\mathcal{I}(t)} &= \{ \langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \\
& \qquad \exists v > t.(\langle d_1, \ldots, d_n \rangle \in R_2^{\mathcal{I}(v)} \wedge \forall w \in (t, v).\, \langle d_1, \ldots, d_n \rangle \in R_1^{\mathcal{I}(w)})\} \\
(R_1\, \mathcal{S}\, R_2)^{\mathcal{I}(t)} &= \{ \langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \\
& \qquad \exists v < t.(\langle d_1, \ldots, d_n \rangle \in R_2^{\mathcal{I}(v)} \wedge \forall w \in (v, t).\, \langle d_1, \ldots, d_n \rangle \in R_1^{\mathcal{I}(w)})\} \\
(\Diamond^+ R)^{\mathcal{I}(t)} &= \{\langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v > t.\, \langle d_1, \ldots, d_n \rangle \in R^{\mathcal{I}(v)}\} \\
(\oplus R)^{\mathcal{I}(t)} &= \{\langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle d_1, \ldots, d_n \rangle \in R^{\mathcal{I}(t+1)}\} \\
(\Diamond^- R)^{\mathcal{I}(t)} &= \{\langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v < t.\, \langle d_1, \ldots, d_n \rangle \in R^{\mathcal{I}(v)}\} \\
(\ominus R)^{\mathcal{I}(t)} &= \{\langle d_1, \ldots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle d_1, \ldots, d_n \rangle \in R^{\mathcal{I}(t-1)}\} \\
\top^{\mathcal{I}(t)} &= \Delta^{\mathcal{I}} \\
EN^{\mathcal{I}(t)} &\subseteq \top^{\mathcal{I}(t)} \\
(\neg E)^{\mathcal{I}(t)} &= \top^{\mathcal{I}(t)} \setminus E^{\mathcal{I}(t)} \\
(E_1 \sqcap E_2)^{\mathcal{I}(t)} &= E_1^{\mathcal{I}(t)} \cap E_2^{\mathcal{I}(t)} \\
(\exists^{\lesseqgtr k}[U_j]R)^{\mathcal{I}(t)} &= \{ d \in \top^{\mathcal{I}(t)} \mid \sharp\{\langle d_1, \ldots, d_n \rangle \in R^{\mathcal{I}(t)} \mid d_j = d\} \lesseqgtr k\} \\
(E_1\, \mathcal{U}\, E_2)^{\mathcal{I}(t)} &= \{ d \in \top^{\mathcal{I}(t)} \mid \exists v > t.(d \in E_2^{\mathcal{I}(v)} \wedge \forall w \in (t, v). d \in E_1^{\mathcal{I}(w)})\} \\
(E_1\, \mathcal{S}\, E_2)^{\mathcal{I}(t)} &= \{ d \in \top^{\mathcal{I}(t)} \mid \exists v < t.(d \in E_2^{\mathcal{I}(v)} \wedge \forall w \in (v, t). d \in E_1^{\mathcal{I}(w)})\}
\end{aligned}
$$

Fig. 1. Syntax and semantics of $\mathcal{DLR}_{\mathcal{US}}$.

to standard relational databases, with the same interpretation of constants and the same domains along time.

As a language for expressing temporal conceptual schemas we use the combination of the propositional temporal logic with *Since* and *Until* and the (non-temporal) description logic $\mathcal{DLR}$ [9]. The resulting $\mathcal{DLR}_{\mathcal{US}}$ *temporal description logic* can be regarded as a rather expressive fragment of the first-order temporal logic $L^{\{\textbf{since, until}\}}$; cf. [14, 19] and Section 3 below; note that $\mathcal{DLR}$ itself is neither in the guarded fragment of FOL nor in the two variable variables fragment of FOL with counting quantifiers.

The basic syntactical types of $\mathcal{DLR}_{\mathcal{US}}$ are *entities* (i.e., unary predicates, also known as *concepts* or *classes*) and n-ary *relations* of arity ≥ 2. Starting from a set of *atomic entities* (denoted by EN), a set of *atomic relations* (denoted by RN), and a set of *role symbols* (denoted by U) we define inductively (complex) entity and relation expressions as is shown in the upper part of Fig. 1, where the binary constructs ($\sqcap, \sqcup, \mathcal{U}, \mathcal{S}$) are applied to relations of the same arity, i, j, k, n are natural numbers, $i \leq n$, and j does not exceed the arity of R.

The non-temporal fragment of $\mathcal{DLR}_{\mathcal{US}}$ coincides with $\mathcal{DLR}$. For both entity and relation expressions all the Boolean constructs are available. The selection expression $U_i/n : E$ denotes an n-ary relation whose argument named U_i ($i \leq n$) is of type E; if it is clear from the context, we omit n and write $(U_i : E)$. The projection expression $\exists^{\lessgtr k}[U_j]R$ is a generalisation with cardinalities of the projection operator over the argument named U_j of the relation R; the plain classical projection is $\exists^{\geq 1}[U_j]R$. It is also possible to use the pure argument position version of the model by replacing role symbols U_i with the corresponding position numbers i.

The language of $\mathcal{DLR}_{\mathcal{US}}$ is interpreted in *temporal models* over $\mathcal{T}$, which are triples of the form $\mathcal{I} \doteq \langle \mathcal{T}, \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}(t)} \rangle$, where $\Delta^{\mathcal{I}}$ is non-empty set of objects (the *domain* of $\mathcal{I}$) and $\cdot^{\mathcal{I}(t)}$ an *interpretation function* such that, for every $t \in \mathcal{T}$, every entity E, and every n-ary relation R, we have $E^{\mathcal{I}(t)} \subseteq \Delta^{\mathcal{I}}$ and $R^{\mathcal{I}(t)} \subseteq (\Delta^{\mathcal{I}})^n$. The semantics of entity and relation expressions is defined in the lower part of Fig. 1, where $(u, v) = \{w \in \mathcal{T} \mid u < w < v\}$ and the operators $\Box^+$ (always in the future) and $\Box^-$ (always in the past) are the duals of $\Diamond^+$ (some time in the future) and $\Diamond^-$ (some time in the past), respectively, i.e., $\Box^+ E \equiv \neg\Diamond^+\neg E$ and $\Box^- E \equiv \neg\Diamond^-\neg E$, for both entities and relations. For entities, the temporal operators $\Diamond^+$, $\oplus$ (at the next moment), and their past counterparts can be defined via $\mathcal{U}$ and $\mathcal{S}$: $\Diamond^+ E \equiv \top\ \mathcal{U}\ E$, $\oplus\ E \equiv \bot\ \mathcal{U}\ E$, etc. However, this is not possible for relations of arity > 1, since $\top_n$—the top n-ary relation—can be interpreted by different subsets of the n-ary cross product $\top \times \cdots \times \top$ at different time points; the reason for this is the ability of $\mathcal{DLR}$ to talk only about difference between relations rather than of the complement of a relation. Then, we may have $\langle d_1, d_2 \rangle \in (\Diamond^+ R)^{\mathcal{I}(t)}$ because $\langle d_1, d_2 \rangle \in R^{\mathcal{I}(t+2)}$, but $\langle d_1, d_2 \rangle \notin (\top_2)^{\mathcal{I}(t+1)}$. The operators $\Diamond^*$ (at some moment) and its dual $\Box^*$ (at all moments) can be defined for both entities and relations as $\Diamond^* E \equiv E \sqcup \Diamond^+ E \sqcup \Diamond^- E$ and $\Box^* E \equiv E \sqcap \Box^+ E \sqcap \Box^- E$, respectively.

A *temporal conceptual database schema* (or a *knowledge base*) is a finite set Σ of $\mathcal{DLR}_{\mathcal{US}}$-formulas. Atomic formulas are formulas of the form $E_1 \sqsubseteq E_2$ and $R_1 \sqsubseteq R_2$, with R_1 and R_2 being relations of the same arity. If φ and ψ are $\mathcal{DLR}_{\mathcal{US}}$-formulas, then so are $\neg\varphi, \varphi \wedge \psi, \varphi \mathcal{U} \psi, \varphi \mathcal{S} \psi$. $E_1 \doteq E_2$ is used as an abbreviation for $(E_1 \sqsubseteq E_2) \wedge (E_2 \sqsubseteq E_1)$, for both entities and relations. The global atomic formula $E_1 \sqsubseteq^* E_2$ is used as an abbreviation for $\Box^*(E_1 \sqsubseteq E_2)$, for both entities and relations. Temporal conceptual database schemas specify the constraints for temporal databases.

Given a formula φ, an interpretation $\mathcal{I}$, and a time point $t \in \mathcal{T}$, the truth-relation $\mathcal{I}, t \models \varphi$ (φ holds in $\mathcal{I}$ at moment t) is defined inductively as follows:

$$
\begin{array}{llll}
\mathcal{I}, t \models E_1 \sqsubseteq E_2 & \text{iff } E_1^{\mathcal{I}(t)} \subseteq E_2^{\mathcal{I}(t)} & \mathcal{I}, t \models \varphi\, \mathcal{U}\, \psi & \text{iff } \exists v > t.(\mathcal{I}, v \models \psi \wedge \\
& & & \quad \forall w \in (t, v).\mathcal{I}, w \models \varphi) \\
\mathcal{I}, t \models R_1 \sqsubseteq R_2 & \text{iff } R_1^{\mathcal{I}(t)} \subseteq R_2^{\mathcal{I}(t)} & \mathcal{I}, t \models \varphi\, \mathcal{S}\, \psi & \text{iff } \exists v < t.(\mathcal{I}, v \models \psi \wedge \\
& & & \quad \forall w \in (v, t).\mathcal{I}, w \models \varphi) \\
\mathcal{I}, t \models \neg\varphi & \text{iff } \mathcal{I}, t \not\models \varphi & & \\
\mathcal{I}, t \models \varphi \wedge \psi & \text{iff } \mathcal{I}, t \models \varphi \text{ and } \mathcal{I}, t \models \psi & &
\end{array}
$$

A formula φ is called *satisfiable* if there is a temporal model $\mathcal{I}$ such that $\mathcal{I}, t \models \varphi$, for some time point t. A conceptual schema Σ is *satisfiable* if the conjunction $\bigwedge \Sigma$ of all formulas in Σ is satisfiable (we write $\mathcal{I}, t \models \Sigma$ instead of $\mathcal{I}, t \models \bigwedge \Sigma$); in this case $\mathcal{I}$ is called a *model* of Σ. We say that Σ is *globally satisfiable* if there is $\mathcal{I}$ such that $\mathcal{I}, t \models \Sigma$ for every t ($\mathcal{I} \models \Sigma$, in symbols). An entity E (or relation R) is *satisfiable* if there is $\mathcal{I}$ such that $E^{\mathcal{I}(t)} \neq \emptyset$ (respectively, $R^{\mathcal{I}(t)} \neq \emptyset$), for some time point t. Finally, we say that Σ (*globally*) *implies* φ and write $\Sigma \models \varphi$ if we have $\mathcal{I} \models \varphi$ whenever $\mathcal{I} \models \Sigma$.

Note that an entity E is satisfiable iff $\neg(E \sqsubseteq \bot)$ is satisfiable. An n-ary relation R is satisfiable iff $\neg(\exists^{\geq 1}[i]R \sqsubseteq \bot)$ is satisfiable for some $i \leq n$. A conceptual schema Σ is globally satisfiable iff $\Box^*(\bigwedge \Sigma)$ is satisfiable. And $\Sigma \models \varphi$ iff $\Box^*(\bigwedge \Sigma) \wedge \neg\varphi$ is not satisfiable. Thus, all reasoning tasks connected with the notions introduced above reduce to satisfiability of formulas.

2.1 Temporal queries

In this Section we extend $\mathcal{DLR}_{\mathcal{US}}$ with a temporal query language, and we define the problem of evaluating a temporal query under $\mathcal{DLR}_{\mathcal{US}}$ constraints and the problem of temporal query containment under constraints (see, e.g., [14, 12, 1] for a survey and a discussion about temporal queries). A *non-recursive Datalog query* (i.e., a disjunction of conjunctive queries or SPJ-queries) over a $\mathcal{DLR}_{\mathcal{US}}$ schema Σ is an expression of the form

$$\mathcal{Q}(\overrightarrow{x}) :- \bigvee_j \mathcal{Q}_j(\overrightarrow{x}, \overrightarrow{y_j}, \overrightarrow{c_j}), \quad \text{where} \quad \mathcal{Q}_j(\overrightarrow{x}, \overrightarrow{y_j}, \overrightarrow{c_j}) \equiv \bigwedge_i P^i_j(\overrightarrow{x}^i_j, \overrightarrow{y}^i_j, \overrightarrow{c}^i_j),$$

P^i_j are $\mathcal{DLR}_{\mathcal{US}}$ entity or relation expressions possibly occurring in Σ, $\overrightarrow{x}^i_j$, $\overrightarrow{y}^i_j$, and $\overrightarrow{c}^i_j$ are sequences of distinguished variables, existential variables, and constants, respectively, the number of which is in agreement with the arity of P^i_j. The variables $\overrightarrow{x}$ in the head are the union of all the distinguished variables in each $\mathcal{Q}_j$; the existential variables are used to make coreferences in the query, and constants are fixed values. $\mathcal{Q}$ is a n-ary relation (not appearing in Σ) whose arity is the number of variables in $\overrightarrow{x}$.

It is to be noted that we allow entities and relations in the query to occur in the conceptual schema Σ. This approach is similar to that of [9], where atoms in a query can be constrained by means of schema formulas. Furthermore, query expressions do not directly manipulate explicit temporal attributes, but time is implicit in each query expression. Indeed, the temporal dimension is handled by means of the temporal modal operators in $\mathcal{DLR}_{\mathcal{US}}$. In this perspective, the query language is in strict relation with the First-Order Temporal Logic with *since* and *until*, $L^{\{\textbf{since, until}\}}$, used in [14] for querying temporal databases.

The semantics of queries is based on the snapshot representation of a temporal database, and defined as follows. Given a temporal schema Σ, let $\mathcal{I}$ be a temporal model, and t be a time point in $\mathcal{T}$ such that $\mathcal{I}$ satisfies Σ at t, i.e., $\mathcal{I}, t \models \Sigma$. The snapshot interpretation

$$\mathcal{I}(t) = \left\langle \Delta^{\mathcal{I}}, \{E^{\mathcal{I}(t)} \mid E \in EN\}, \{R^{\mathcal{I}(t)} \mid R \in RN\} \right\rangle$$

can be regarded as a usual first-order structure (i.e., a snapshot, non-temporal, database at time t conforming in a sense to the conceptual schema), and so the whole $\mathcal{I}$ as a first-order temporal model (with constant domain $\Delta^{\mathcal{I}}$ in which some values of the query constants are specified). The *evaluation* of a query $\mathcal{Q}$ of arity n, under the constraints Σ, in the model $\mathcal{I}$ that satisfies Σ at moment t, and the *answer* to the query $\mathcal{Q}$, are respectively the sets:

$$\mathsf{eval}(\mathcal{Q}, \mathcal{I}(t)) = \{\overrightarrow{o} \in (\Delta^{\mathcal{I}})^n \mid \mathcal{I}, t \models \bigvee_j \exists \overrightarrow{y_j}. \mathcal{Q}_j(\overrightarrow{o}, \overrightarrow{y_j}, \overrightarrow{c_j})\}$$
$$\mathsf{ans}(\mathcal{Q}, \mathcal{I}) = \{\langle t, \overrightarrow{o}\rangle \in \mathcal{T} \times (\Delta^{\mathcal{I}})^n \mid \overrightarrow{o} \in \mathsf{eval}(\mathcal{Q}, \mathcal{I}(t))\}$$

We obtain a so called *sequenced semantics* for queries, which is based on the view of a database as a time-indexed collection of snapshots. The query language is also *snapshot-reducible* in the sense that non-temporal queries—i.e., queries without any temporal connective—are still valid queries, and are interpreted using the sequenced semantics. Our language allows also for *upward compatible* queries. Intuitively, a non-temporal query is upward compatible if the answer set on a temporal database is the same as the answer set on an associated non-temporal database. This is a temporal *slice* of the temporal database at the current time: $\mathsf{eval}(\mathcal{Q}, \mathcal{I}(\mathsf{now}))$.

Given two queries (of the same arity) $\mathcal{Q}_1$ and $\mathcal{Q}_2$ over Σ, we say that $\mathcal{Q}_1$ is *contained* in $\mathcal{Q}_2$ under the constraints Σ, and write $\Sigma \models \mathcal{Q}_1 \subseteq \mathcal{Q}_2$, if, for every temporal model $\mathcal{I}$ of Σ we have $\mathsf{ans}(\mathcal{Q}_1, \mathcal{I}) \subseteq \mathsf{ans}(\mathcal{Q}_2, \mathcal{I})$. The *query satisfiability problem*—given a

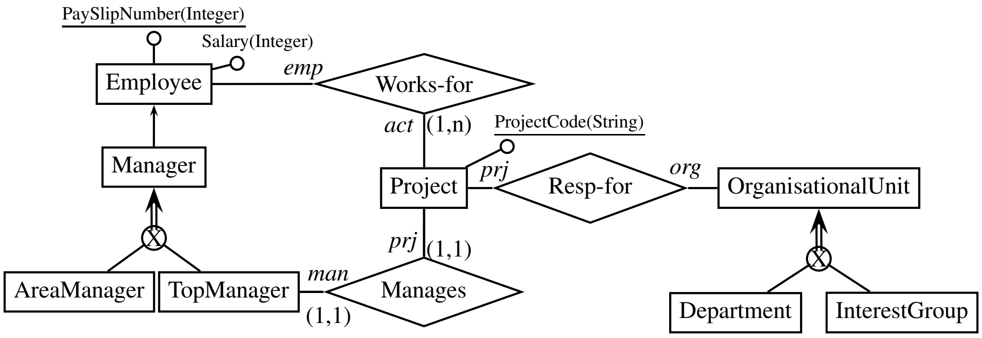

Fig. 2. The example EER diagram.

query $\mathcal{Q}$ over a schema Σ, to determine whether there are $\mathcal{I}$ and t such that $\mathcal{I}, t \models \Sigma$, and $\mathsf{eval}(\mathcal{Q}, \mathcal{I}(t)) \neq \emptyset$—is reducible to query containment: $\mathcal{Q}$ is satisfiable iff $\Sigma \not\models \mathcal{Q}(\overrightarrow{x}) \subseteq P(\overrightarrow{x}) \wedge \neg P(\overrightarrow{x})$, where P is a $\mathcal{DLR}_{\mathcal{US}}$-relation of the same arity as $\mathcal{Q}$.

2.2 Examples

As an example, let us consider the following conceptual schema Σ:

$$
\begin{array}{l}
\mathtt{Works\text{-}for} \sqsubseteq^* \mathtt{emp/2} : \mathtt{Employee} \sqcap \mathtt{act/2} : \mathtt{Project} \\
\mathtt{Manages} \sqsubseteq^* \mathtt{man/2} : \mathtt{TopManager} \sqcap \mathtt{prj/2} : \mathtt{Project} \\
\mathtt{Employee} \sqsubseteq^* \exists^{=1}[\mathtt{from}]\mathtt{PaySlipNumber} \sqcap \exists^{=1}[\mathtt{from}](\mathtt{PaySlipNumber} \sqcap \mathtt{to/2} : \mathtt{Integer}) \sqcap \\
\qquad\qquad \exists^{=1}[\mathtt{from}]\mathtt{Salary} \sqcap \exists^{=1}[\mathtt{from}](\mathtt{Salary} \sqcap \mathtt{to/2} : \mathtt{Integer}) \\
\top \sqsubseteq^* \exists^{\leq 1}[\mathtt{to}](\mathtt{PaySlipNumber} \sqcap \mathtt{from/2} : \mathtt{Employee}) \\
\mathtt{Manager} \sqsubseteq^* \mathtt{Employee} \sqcap (\mathtt{AreaManager} \sqcup \mathtt{TopManager}) \\
\mathtt{AreaManager} \sqsubseteq^* \mathtt{Manager} \sqcap \neg\mathtt{TopManager} \\
\mathtt{TopManager} \sqsubseteq^* \mathtt{Manager} \sqcap \exists^{=1}[\mathtt{man}]\mathtt{Manages} \\
\mathtt{Project} \sqsubseteq^* \exists^{\geq 1}[\mathtt{act}]\mathtt{Works\text{-}for} \sqcap \exists^{=1}[\mathtt{prj}]\mathtt{Manages} \\
\mathtt{Employee} \sqcap \neg(\exists^{\geq 1}[\mathtt{emp}]\mathtt{Works\text{-}for}) \sqsubseteq^* \mathtt{Manager} \\
\mathtt{Manager} \sqsubseteq^* \neg(\exists^{\geq 1}[\mathtt{emp}]\mathtt{Works\text{-}for}) \sqcap (\mathtt{Qualified}\ \mathcal{S}\ (\mathtt{Employee} \sqcap \neg\mathtt{Manager}))
\end{array}
$$

The theory introduces `Works-for` as a binary relation between employees and projects, and `Manages` as a binary relation between managers and projects. Employees have exactly one pay slip number and one salary each, which are represented as binary relations (with `from` and `to` roles) with an integer domain; moreover, a pay slip number uniquely identifies an employee (it acts as a key). It is stated that managers are employees, and are partitioned into area managers and top managers. Top Managers participate exactly once in the relation `Manages`, i.e., every top manager manages exactly one project. Projects participate at least once to the relation `Works-for` and exactly once in the relation `Manages`. Finally, employees not working for a project are exactly the managers, and managers should be qualified, i.e., should have passed a period of being employees. The meaning of the above conceptual schema (with the exception of the last two formulas) is illustrated by the left part of the diagram in Fig. 2.

The conceptual schema Σ globally logically implies that, for every project, there is at least one employee who is not a manager, and that a top manager worked in a project before managing some (possibly different) project:

$\Sigma \models \texttt{Project} \sqsubseteq^* \exists^{\geq 1}[\texttt{act}](\texttt{Works-for} \sqcap \texttt{emp} : \neg\texttt{Manager})$
$\Sigma \models \texttt{TopManager} \sqsubseteq^* \Diamond^- \exists^{\geq 1}[\texttt{emp}](\texttt{Works-for} \sqcap \texttt{act} : \texttt{Project})$

Note also that if we add to Σ the formula

$\texttt{Employee} \sqsubseteq^* \exists^{\geq 1}[\texttt{emp}]\texttt{Works-for}$

saying that every employee should work for at least one project, then all the entities and the relations mentioned in the conceptual schema are interpreted as the empty set in every model of Σ, i.e., they are not satisfiable relative to Σ.

The expressivity of the query language can be understood with the following examples, taken from [14]:

"*Find all people who have worked for only one project*"
$\mathcal{Q}(\texttt{x}) :- (\exists^{=1}[\texttt{emp}](\Diamond^*\texttt{Works-for}))(\texttt{x})$
"*Find all managers whose terminal project has code* prj342"
$\mathcal{Q}(\texttt{x}) :- \texttt{Manager}(\texttt{x}) \wedge \texttt{Manages}(\texttt{x}, \mathsf{prj342}) \wedge (\Box^+\neg\texttt{Manages})(\texttt{x}, \texttt{y})$
"*Find all project-hoppers—people who never spent more than two consecutive years in a project*"
$\mathcal{Q}(\texttt{x}) :- (\Box^*\neg\exists^{\geq 1}[\texttt{emp}](\texttt{Works-for} \sqcap \oplus\, \texttt{Works-for} \sqcap \oplus \oplus\, \texttt{Works-for}))(\texttt{x})$
"*Find all people who did not work between two projects*"
$\mathcal{Q}(\texttt{x}) :- (\Diamond^-\exists^{\geq 1}[\texttt{emp}]\texttt{Works-for})(\texttt{x}) \wedge$
$\quad (\neg\exists^{\geq 1}[\texttt{emp}]\texttt{Works-for})(\texttt{x}) \wedge (\Diamond^+\exists^{\geq 1}[\texttt{emp}]\texttt{Works-for})(\texttt{x})$

We now consider an example of query containment under constraints, where the constraints are expressed by the above schema Σ. Consider the following queries:

$\mathcal{Q}_1(\texttt{x}, \texttt{y}) :- \neg\texttt{AreaManager}(\texttt{x}) \wedge \texttt{Manages}(\texttt{x}, \texttt{z}) \wedge \texttt{Project}(\texttt{z}) \wedge$
$\quad \texttt{Resp-for}(\texttt{y}, \texttt{z}) \wedge \texttt{Department}(\texttt{y})$
$\mathcal{Q}_2(\texttt{x}, \texttt{y}) :- (\Diamond^-\exists^{\geq 1}[\texttt{emp}]\texttt{Works-for})(\texttt{x}) \wedge \texttt{Manages}(\texttt{x}, \texttt{z}) \wedge$
$\quad \texttt{Resp-for}(\texttt{y}, \texttt{z}) \wedge \neg\texttt{InterestGroup}(\texttt{y})$

It is not hard to see that $\mathcal{Q}_1$ is contained in $\mathcal{Q}_2$ under the constraints in Σ, i.e., $\Sigma \models \mathcal{Q}_1 \subseteq \mathcal{Q}_2$.

3 Decidability and complexity

In this section, we present and briefly discuss our main results on the computational behaviour of $\mathcal{DLR}_{\mathcal{US}}$ and its fragments over the flow of time $\langle \mathbb{Z}, < \rangle$.

Unfortunately, full $\mathcal{DLR}_{\mathcal{US}}$, even restricted to *atomic* formulas, turns out to be undecidable. One can actually show that the undecidable tiling problem reduces to satisfiability of atomic formulas in $\mathcal{DLR}_{\mathcal{US}}$.

Theorem 1. *The global satisfiability problem for $\mathcal{DLR}_{\mathcal{US}}$ conceptual schemas containing only atomic formulas is undecidable.*

It follows, in particular, that (a) the problem of satisfiability of complex $\mathcal{DLR}_{\mathcal{US}}$ formulas is undecidable, and (b) the problem of global logical implication in $\mathcal{DLR}_{\mathcal{US}}$—even involving only atomic formulas—is undecidable as well. The main technical reason for undecidability is the possibility to 'temporalise' binary relations, cf. [26]. The fragment $\mathcal{DLR}^-_{\mathcal{US}}$, in which the temporal operators can be applied only to entities and formulas (but not to n-ary relations, with $n \geq 2$), exhibits a much better computational behaviour. The following theorem presents the complexity results we have obtained for schema and query reasoning in $\mathcal{DLR}^-_{\mathcal{US}}$:

Theorem 2. *Let the flow of time be $\langle \mathbb{Z}, < \rangle$. Then*

(1) *the problem of logical implication in $\mathcal{DLR}_{\mathcal{US}}^{-}$ involving only atomic formulas is EXPTIME-complete;*

(2) *the formula satisfiability problem (and so the problem of logical implication) in $\mathcal{DLR}_{\mathcal{US}}^{-}$ is EXPSPACE-complete;*

(3) *the query-containment problem for non-recursive Datalog queries under $\mathcal{DLR}_{\mathcal{US}}^{-}$-constraints is decidable in 2EXPTIME and is EXPSPACE-hard.*

The main ideas of the proof are as follows:

(1) EXPTIME-hardness follows from the EXPTIME-hardness of $\mathcal{DLR}$. An EXPTIME-algorithm is obtained by means of a polynomial reduction from $\mathcal{DLR}_{\mathcal{US}}^{-}$ to the logic $\mathcal{DLR}_{reg}$, where logical implication is known to be decidable in EXPTIME [9]. The reduction extends the one proposed in [21].

(2)The EXPSPACE-hardness of the formula satisfiability problem is proved in [17], even for the sublanguage $\mathcal{ALC}_{\Box}$ of $\mathcal{DLR}_{\mathcal{US}}^{-}$, by reducing to it the *n*-CORRIDOR tiling problem [24]. But the most interesting and original result presented in this paper is the EXPSPACE upper bound. We briefly sketch the main new contribution. First, we take the road proposed in [25, 27, 28] and show that the satisfiability problem can be equivalently formulated as a problem about the existence of certain *quasimodels*. Roughly, the idea behind the notion of a quasimodel is to represent 'the state' of the (in general, infinite) domain of a temporal model at each moment of time by finitely many 'types' of the domain objects at this moment. A set T is a *quasistate candidate* for φ if it is a set of concept types (i.e., a subset of all the subconcepts in φ which is Boolean saturated). Not all quasistate candidates can represent proper states. Denote by $c(t)$ the type which results from t when all concepts starting with a temporal operator are replaced by new atomic concepts. Thus, $c(t)$ abstracts from the temporal content of T. We say that the quasistate candidate T is a *quasistate* for φ if the following (non-temporal) $\mathcal{DLR}$-formula α_T

$$\big(\bigsqcup_{t \in T} c(t) \doteq \top\big) \wedge \bigwedge_{t \in T} \neg(c(t) \doteq \bot)$$

is satisfiable. A *quasimodel* for φ is a sequence of quasistates for φ satisfying some additional conditions. It was proved in [28] that given an oracle deciding whether a given quasistate candidate is a quasistate, the question whether a quasimodel for φ exists can be decided in EXPSPACE. Hence, the main new ingredient providing us with an EXPSPACE procedure deciding the existence of quasimodels for φ is the following:

Lemma 1. *Given a $\mathcal{DLR}_{\mathcal{US}}^{-}$-formula φ, it is decidable in EXPSPACE whether a quasistate candidate for φ is a quasistate.*

Since α_T is exponential in the size of φ, this lemma does not follow from known results and, in fact, a number of new ideas are required to prove it.

(3) The query containment problem can be reduced to satisfiability in quasimodels and the query-containment problem for (non-temporal) $\mathcal{DLR}$. Since we proved in (2) that the satisfiability problem is EXPSPACE-complete, while for $\mathcal{DLR}$ the problem was shown to be decidable in 2EXPTIME time in [9] we can conclude that also in $\mathcal{DLR}_{\mathcal{US}}^{-}$ the query containment problem is in 2EXPTIME.

4 Conceptual modelling

In this section we briefly show how the temporal description logic $\mathcal{DLR}_{\mathcal{US}}$ can provide a formal semantic characterisation of the most important temporal conceptual modelling constructs (for the valid time representation). We refer mostly to the temporal extended entity-relationship data model, for which a detailed literature exists [18, 22].

The extended entity-relationship (EER) model—i.e., the standard entity-relationship data model, enriched with IsA links, disjoint and covering constraints, and full cardinality constraints—may be viewed as a temporalised EER model which assigns to every construct a temporal interpretation but provides no explicit temporal constructs. Gregersen and Jensen [18] call this approach *implicit*, because the temporal dimension is hidden in the interpretation structure so that entities and relationships are always time-dependent. The non-temporal fragment of $\mathcal{DLR}_{\mathcal{US}}$, i.e., $\mathcal{DLR}$, is enough to capture the EER model with implicit time. For the non-temporal EER model, such an encoding, introduced by [10, 11], establishes a precise correspondence between legal database states of the EER diagram and models of the derived $\mathcal{DLR}$ theory. That this encoding is correct for the EER model with implicit time was shown in [2, 3], where $\mathcal{DLR}$ was interpreted by a temporal semantics. The example knowledge base in Section 2.2 (without the last two formulas) shows the exact encoding for the left-hand part of the temporally implicit EER diagram in Fig.2. This encoding could support the design of a temporal conceptual schema by exploiting the reasoning in $\mathcal{DLR}_{\mathcal{US}}$: it becomes possible to verify the conceptual specification, infer implicit facts and stricter constraints, and manifest any inconsistencies. Note that the same ideas presented here would apply to non-temporal UML class diagrams (encoded in $\mathcal{DLR}$ in [8]) and DAML+OIL ontologies.

We now introduce the basic temporal constructs that can be added on top of a temporally implicit model leading to a full fledged temporal conceptual model.

Temporal entities and relations. Both entity and relation instances in a temporal setting have an existence time associated to them. $\mathcal{DLR}_{\mathcal{US}}$-formulas can enforce either that entities (relations) cannot last forever—we call them *temporary entities* (*temporary relations*), or that their extension never changes in time—we call them *snapshot entities* (*snapshot relations*). Temporary entities and relations are captured by the following $\mathcal{DLR}_{\mathcal{US}}$-formulas:

$$E \sqsubseteq^* (\Diamond^+ \neg E) \sqcup (\Diamond^- \neg E) \qquad\qquad R \sqsubseteq^* (\Diamond^+ \neg R) \sqcup (\Diamond^- \neg R)$$

saying that there must be a past or a future time point where the entity (relation) does not hold. In other words, instances of temporary entities (relations) always have a limited lifetime. On the other hand, snapshot entities and relations are captured by the following $\mathcal{DLR}_{\mathcal{US}}$-formulas:

$$E \sqsubseteq^* (\Box^+ E) \sqcap (\Box^- E) \qquad\qquad R \sqsubseteq^* (\Box^+ R) \sqcap (\Box^- R)$$

saying that whenever the entity (relation) is true it is necessarily true in every past and every future time point, i.e, they never change along time. Snapshot entities and relations are used to capture the semantics of legacy non-temporal schemas when included in a temporal model, thus enforcing the upward compatibility. In our example (Fig. 2), `Employee,Department,Resp-For` could be constrained by snapshot $\mathcal{DLR}_{\mathcal{US}}$ formulas, while `Manager,Work-For` by temporary formulas.

Temporal attributes. At different points in time, an entity may have different values for the same attribute. Attributes can be forced either to remain unchanged in time (*snapshot* attribute), or to necessarily change (*temporary* attribute) with the following $\mathcal{DLR}_{\mathcal{US}}$-formulas, respectively:

$$E \sqsubseteq^* \exists^{\geq 1}[\mathtt{from}]\Box^* R_A \qquad\qquad E \sqsubseteq^* \exists^{=1}[\mathtt{From}](R_A \sqcap (\Diamond^+ \neg R_A \sqcup \Diamond^- \neg R_A))$$

When considering the interaction between the temporal behaviour of an attribute and that of the owner entity, it is consistent in $\mathcal{DLR}_{\mathcal{US}}$ to have both snapshot attributes of a temporary entity and temporary attributes of a snapshot entity. In the former case, the $\mathcal{DLR}_{\mathcal{US}}$ semantics says that during the lifespan of an entity the value of a snapshot attribute never changes. In the latter one, the meaning is that each instance always belongs to the snapshot entity but the value of the temporary attribute will change during its existence. In our running example, where `Employee` is a snapshot entity, `Salary` is modelled as a temporary attribute, while `Names,PaySlipNumber` are modelled as snapshot attributes. In particular, `PaySlipNumber` plays the role of a full fledged temporal key (see the formula for `PaySlipNumber` in Sect. 2.2).

Temporal cardinalities. Cardinality constraints limit the participation of entities in relationships. In a temporal setting, we can distinguish between *snapshot participation constraints* (true at each point in time) and *lifespan participation constraints* (evaluated during the entire existence of the entity). While the standard $\mathcal{DLR}_{\mathcal{US}}$ cardinality construct captures snapshot participation constraints, the lifespan participation constraints are defined by the following $\mathcal{DLR}_{\mathcal{US}}$-formula:

$$E \sqsubseteq^* \exists^{\lessgtr n}[i]\Diamond^* R$$

i.e., over its lifespan, an instance of the entity E must participate as the ith argument in at least n (at most n, or precisely n) tuples of the relation R. Obviously, since for snapshot relations the set of instances does not change in time, there is no difference between snapshot and lifespan participation constraints with respect to snapshot relations. In our example, we could say for example that managers should manage at most 5 different projects in their entire existence as managers, while still being constrained in managing exactly one project at a time.

Dynamic entities. In the temporal conceptual modelling literature, two notions of dynamic transitions between a entities (also called object migrations) are considered [22]: *dynamic evolution*, when an object ceases to be an instance of a source entity, and *dynamic extension*, when an object continues to belong to the source. Let E_s be the source entity and E_t be the target one; the two cases are captured by the following formulas, respectively:

$$E_s \sqsubseteq^* \Diamond^+(E_t \sqcap \neg E_s) \qquad\qquad E_s \sqsubseteq^* \Diamond^+ E_t$$

An interesting consequence of dynamic evolution is that the source is necessarily a temporary entity.

$\mathcal{DLR}_{\mathcal{US}}$ is also able to capture *safety* and *liveness* constraints. A *safety* constraint intuitively says that "*nothing bad ever happens*" and can be captured by the formula $\Box^*\neg(E \sqsubseteq \bot)$. On the other hand, *liveness* constraints saying that "*something good will happen*" can be expressed by existential temporal formulas: $\Diamond^*\neg(E \sqsubseteq \bot)$.

Schema evolution. We consider here a simplified case of conceptual schema evolution [15], which can be called a *monotonic* approach. This allows only for changes in the schema such that the resulting conceptual schema is compatible with the previous *global* constraints.

Let Γ and ϕ_p be atomic formulas, or possibly a Boolean combination of atomic formulas, which introduce, respectively, a new schema portion and a condition to be checked. The formula $\Box^*(\neg\phi_p \vee \Box^+\Gamma)$ states that as soon as the property ϕ_p becomes true for the data, the conceptual schema will include the additional Γ constraint. A simple example is:

$\phi_{\mathtt{p}} \equiv (\mathtt{InterestGroup} \sqsubseteq \bot)$
$\Gamma \equiv (\exists^{\geq 1}[\mathtt{amount}](\mathtt{Salary} \sqcap \mathtt{payee/2} : \mathtt{TopManager}) \sqsubseteq \mathtt{LowAmount})$

meaning that as soon as the organisation does not include interest groups anymore, the salary of top managers should be in the "low amount" class.

Note that the $\mathcal{DLR}^{-}_{\mathcal{US}}$ with only atomic formulas (i.e., the EXPTIME-complete fragment of $\mathcal{DLR}_{\mathcal{US}}$) is enough to capture most of the modelling constructs discussed in this Section—in fact, in the case of (global) logical implication for atomic formulas, there is no difference between $\sqsubseteq$ and $\sqsubseteq^*$—with the exception of (a) schema evolution constraints, (b) safety and liveness conditions, (c) snapshot relations and attributes, and (d) temporal cardinalities. Full $\mathcal{DLR}^{-}_{\mathcal{US}}$ (i.e., the EXPSPACE-complete fragment of $\mathcal{DLR}_{\mathcal{US}}$) is able to express (a) and (b) as well. However, (c) and (d) require temporalised relations which, by Theorem 1, lead to undecidability.

5 Conclusion

This work introduces the temporal description logic $\mathcal{DLR}_{\mathcal{US}}$ and illustrates its expressive power. A temporal query language was defined and the problem of query containment under the constraints defined by a $\mathcal{DLR}_{\mathcal{US}}$ conceptual schema is investigated.

Tight complexity results were proved. In particular, reasoning in the full logic $\mathcal{DLR}_{\mathcal{US}}$ was shown to be undecidable, while decidability was obtained using a still expressive fragment, $\mathcal{DLR}^{-}_{\mathcal{US}}$. We have also shown that the problem of checking query containment of non-recursive Datalog queries under constraints with arbitrary $\mathcal{DLR}^{-}_{\mathcal{US}}$ formulas is decidable in 2EXPTIME with an EXPSPACE lower bound. This result is the first decidability result we are aware of on containment of temporal conjunctive queries under expressive constraints.

This work has been partially funded by the EPSRC grants GR/R45369/01, GR/R04348/01, GR/R09428/01. The second author wishes to thank the University of Manchester, where most of the work presented in this paper was carried out.

References

1. S. Abiteboul, L. Herr, and J. Van den Bussche. Temporal versus first-order logic to query temporal databases. In *Proc. of the 15th ACM SIGACT SIGMOD SIGART Sym. on Principles of Database Systems (PODS'96)*, pages 49–57, 1996.
2. A. Artale and E. Franconi. Reasoning with enhanced temporal entity-relationship models. In *Proc. of the International Workshop on Spatio-Temporal Data Models and Languages*. IEEE Computer Society Press, August 1999.
3. A. Artale and E. Franconi. Temporal ER modeling with description logics. In *Proc. of the International Conference on Conceptual Modeling (ER'99)*. Springer-Verlag, November 1999.
4. A. Artale and E. Franconi. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence*, 30(1-4), 2001.
5. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2002.
6. A. Borgida and R. J. Brachman. Conceptual modelling with description logics. In Baader et al. [5].
7. A. Borgida, M. Lenzerini, and R. Rosati. Description logics for databases. In Baader et al. [5].

8. A. Calì D. Calvanese, G. De Giacomo, and M. Lenzerini. A formal framework for reasoning on UML class diagrams. In *Proc. of the 13th Int. Sym. on Methodologies for Intelligent Systems (ISMIS 2002)*, 2002.
9. D. Calvanese, G. De Giacomo, and M. Lenzerini. On the decidability of query containment under constraints. In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Sym. on Principles of Database Systems (PODS'98)*, pages 149–158, 1998.
10. D. Calvanese, M. Lenzerini, and D. Nardi. Description logics for conceptual data modeling. In Chomicki and Saake [13].
11. D. Calvanese, M. Lenzerini, and D. Nardi. Unifying class-based representation formalisms. *J. of Artificial Intelligence Research*, 11:199–240, 1999.
12. J. Chomicki. Temporal query languages: a survey. In *Proc. of the 1st International Conference on Temporal Logic (ICTL'94)*, pages 506–534, 1994.
13. J. Chomicki and G. Saake, editors. *Logics for Databases and Information Systems*. Kluwer, 1998.
14. J. Chomicki and D. Toman. Temporal logic in information systems. In Chomicki and Saake [13], chapter 1.
15. E. Franconi, F. Grandi, and F. Mandreoli. A semantic approach for schema evolution and versioning in object-oriented databases. In *Proc. of the 1st International Conf. on Computational Logic (CL'2000), DOOD stream*. Springer-Verlag, July 2000.
16. E. Franconi and G. Ng. The ICOM tool for intelligent conceptual modelling. In *Proc. of the 7^{th} International Workshop on Knowledge Representation meets Databases (KRDB'2000)*, 2000.
17. D. Gabbay, A.Kurucz, F. Wolter, and M. Zakharyaschev. *Many-dimensional modal logics: theory and applications*. Studies in Logic. Elsevier, 2002. To appear.
18. H Gregersen and J. S. Jensen. Temporal Entity-Relationship models - a survey. *IEEE Transactions on Knowledge and Data Engineering*, 11(3):464–497, 1999.
19. I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable fragments of first-order temporal logics. *Annals of Pure and Applied Logic*, 106:85–134, 2000.
20. M. Jarke, C. Quix, D. Calvanese, M. Lenzerini, E. Franconi, S. Ligoudistiano, P. Vassiliadis, and Y. Vassiliou. Concept based design of data warehouses: The DWQ demonstrators. In *2000 ACM SIGMOD Intl. Conference on Management of Data*, 2000.
21. K. Schild. Combining terminological logics with tense logic. In *Proceedings of the 6^{th} Portuguese Conference on Artificial Intelligence, EPIA'93*, October 1993.
22. S. Spaccapietra, C. Parent, and E. Zimanyi. Modeling time from a conceptual perspective. In *Int. Conf. on Information and Knowledge Management (CIKM98)*, 1998.
23. H. Sturm and F. Wolter. A tableau calculus for temporal description logic: the expanding domain case. *Journal of Logic and Computation*, 2002. To appear.
24. P. van Emde Boas. The convenience of tiling. Technical Report CT-96-01, ILLC—Univ. of Amsterdam, 1996.
25. F. Wolter and M. Zakharyaschev. Satisfiability problem in description logics with modal operators. In *Proc. of the 6^{th} International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 512–523, Trento, Italy, June 1998.
26. F. Wolter and M. Zakharyaschev. Modal description logics: Modalizing roles. *Fundamenta Informaticae*, 39(4):411–438, 1999.
27. F. Wolter and M. Zakharyaschev. Multi-dimensional description logics. In *Proc. of IJCAI'99*, pages 104–109, 1999.
28. F. Wolter and M. Zakharyaschev. Temporalizing description logics. In D. Gabbay and M. de Rijke, editors, *Frontiers of Combining Systems*, pages 379–401. Studies Press-Wiley, 1999.