

Containment of Conjunctive LTL Queries

Jean Christoph Jung¹, Vladislav Ryzhikov², Frank Wolter³ and Michael Zakharyashev²

¹*TU Dortmund University, Otto-Hahn-Straße 12, 44227 Dortmund, Germany*

²*Birkbeck, University of London, Malet Street, London WC1E 7HX, UK*

³*University of Liverpool, Ashton Street, Liverpool L69 3BX, UK*

Abstract

We investigate the computational complexity of the containment problem for conjunctive queries given in linear temporal logic *LTL* with the operators ‘eventually’ and ‘next-time’. We show that this problem is coNP-complete and identify a few restricted query classes for which containment is tractable (in LOGSPACE).

Keywords

Linear temporal logic LTL, conjunctive query, query containment, computational complexity.

1. Introduction


Our concern in this paper is the containment problem for very basic temporal conjunctive queries that are built from atoms—propositional variables representing atomic events and \top —using conjunction and the temporal operators \bigcirc (at the next moment) and \diamond (sometime later) from the linear temporal logic *LTL*. Such queries are supposed to be evaluated over data instances that consist of facts of the form $A(\ell)$ stating that atomic proposition A is true at time ℓ , for $\ell \in \mathbb{N}$. This setting is relevant to applications in numerous areas ranging from temporal databases and streams to temporal ontology-based data access, pattern matching and learning; see the *Motivation and Related Work* section below for some details and references.

Using the fact that our queries correspond to tree-shaped conjunctive queries, it is readily seen that the *query evaluation problem*—given a query q , a data instance \mathcal{D} , and a timestamp ℓ from the temporal domain of \mathcal{D} , decide whether q is true at ℓ in the model determined by \mathcal{D} —is decidable in polynomial time (in the size of \mathcal{D} and q). The containment problem for these queries turns out to be more interesting.

Recall that the *query containment problem* is to decide, given any queries q and q' , whether, for every data instance \mathcal{D} , the answers to q over \mathcal{D} are contained in the answers to q' over \mathcal{D} . Our main result in this paper proves that query containment for our most expressive query language is non-tractable and coNP-complete. It is the hardness part of the result that is of interest while the upper bound is more or less straightforward. We also show that, for queries without \bigcirc , containment is in the complexity class L (LOGSPACE); it is also in L for path queries with both \bigcirc and \diamond .

As we are only considering queries that are existential *LTL*-formulas (without the operator ‘always in the future’), the containment problem is equivalent to the validity of formulas of the form $q \rightarrow q'$ in finite or infinite temporal models. So, our results also contribute directly to the research problem of classifying fragments of *LTL* according to their computational complexity; see, e.g., [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].

Motivation and Related Work. Temporal data is ubiquitous in the digital world, and there are numerous applications having our non-relational propositional *LTL*-queries as a core. For example, temporal variants of SQL tailored to effectively retrieve information from temporal databases rely on *LTL*-operators [11, 12]. In ontology-based data access (OBDA), temporal conjunctive formulas are often

 DL 2025: 38th International Workshop on Description Logics, September 3–6, 2025, Opole, Poland

 jean.jung@tu-dortmund.de (J. C. Jung); vlad@dcs.bbk.ac.uk (V. Ryzhikov); wolter@liverpool.ac.uk (F. Wolter); michael@dcs.bbk.ac.uk (M. Zakharyashev)

 0000-0002-4159-2255 (J. C. Jung); 0000-0002-6847-6465 (V. Ryzhikov); 0000-0002-4470-606X (F. Wolter); 0000-0002-2210-5183 (M. Zakharyashev)



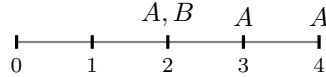
© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

utilised as a core user-friendly language to formulate both queries and ontology rules; see, e.g., [13] and further references therein as well as more recent [14, 15, 16]. Another recent application of *LTL*-queries is extraction of (or learning) patterns from temporal data examples [17]. In this setting, conjunctive *LTL*-queries, their subclasses such as path queries and extensions, say with disjunction, have been studied in [18, 19, 20, 21]. Note also that there is a close link between evaluating path queries and algorithms for finding patterns in strings [22, 23], a problem having multiple applications such as DNA analysis in bioinformatics.

Query containment was shown to be NP-complete for standard database conjunctive queries by reduction to query evaluation in the seminal work [24]. Since then query containment has been studied for numerous query languages. Most closely related is work on containment for fragments of XPath [25] and query containment over trees [26]. In these languages, similar to our work, queries can refer to nodes reachable along the transitive closure of a successor relation. Very different techniques are used, however, since we assume a linear order instead of a tree. Our results are also closely related to the work on subsumption in extensions of \mathcal{EL} . Observe that our queries are notational variants of \mathcal{EL} -concepts with two roles (one corresponding to \bigcirc and the other to \Diamond). Hence, subsumption between \mathcal{EL} concepts over interpretations, in which one role is interpreted as the transitive closure of another role, corresponds exactly to query containment in our language, except that arbitrary interpretations (equivalently, by unfolding, tree-shaped ones) are considered, and not linear orders. In [27], coNP-hardness is shown for subsumption in \mathcal{EL} with transitive closure using the technique introduced for XPath in [25]. Again, this technique is not applicable for linear orders.

2. Temporal Data and Queries

Temporal data instances. By a *temporal data instance* we mean here any finite set $\mathcal{D} \neq \emptyset$ of *facts* of the form $A(\ell)$, where A is a propositional variable, henceforth called an *atom*, and $\ell \in \mathbb{N}$. Intuitively, $A(\ell)$ says that event A happened at time instant ℓ . The *signature* $\text{sig}(\mathcal{D})$ of \mathcal{D} is the set of atoms occurring in it. The maximal time ℓ in \mathcal{D} is denoted by $\max \mathcal{D}$. Where convenient, we also write \mathcal{D} as the word $\delta_0 \dots \delta_{\max \mathcal{D}}$ over the alphabet $2^{\text{sig}(\mathcal{D})}$ with $\delta_i = \{A \mid A(i) \in \mathcal{D}\}$ for each $i \leq \max \mathcal{D}$. By definition, $\delta_{\max \mathcal{D}} \neq \emptyset$. For example, the data instance $\mathcal{D} = \{A(2), B(2), A(3), A(4)\}$, illustrated in the picture below, is also given by the word $\emptyset\emptyset\{A, B\}\{A\}\{A\}$ or $\emptyset^2\{A, B\}\{A\}^2$ if we use standard formal language abbreviations.



Temporal conjunctive queries. To query data instances, we employ *LTL*-formulas that are built from atoms and the logical constant \top using conjunction \wedge and the unary temporal operators \bigcirc (next time) and \Diamond (sometime in the future). We consider the following classes of queries:

$\mathcal{Q}[\bigcirc\Diamond]$: all $\bigcirc\Diamond$ -queries (that is, all queries in our language);

$\mathcal{Q}[\Diamond]$: all \Diamond -queries (that is, $\bigcirc\Diamond$ -queries not containing \bigcirc);

$\mathcal{Q}_p[\bigcirc\Diamond]$: *path* $\bigcirc\Diamond$ -queries, that is, $\bigcirc\Diamond$ -queries of the form

$$q = \rho_0 \wedge \mathbf{o}_1 (\rho_1 \wedge \mathbf{o}_2 (\rho_2 \wedge \dots \wedge \mathbf{o}_{n-1} (\rho_{n-1} \wedge \mathbf{o}_n \rho_n))), \quad (1)$$

where $\mathbf{o}_i \in \{\bigcirc, \Diamond\}$ and the ρ_i are conjunctions of atoms (we often treat these conjunctions as sets and the empty conjunction as \top);

$\mathcal{Q}_p[\Diamond]$: *path* \Diamond -queries, that is, those queries in $\mathcal{Q}_p[\bigcirc\Diamond]$ that do not contain \bigcirc ;

\mathcal{Q}_{in} : *interval-queries*, that is, queries of the form (1) with $\rho_0 = \top$, $\rho_1 \neq \top$, $\mathbf{o}_1 = \Diamond$, and $\mathbf{o}_i = \bigcirc$, for $i > 1$.

We do not consider query classes with \bigcirc -operators only, for which the containment problem is trivial. The *truth-relation* $\mathcal{D}, \ell \models q$, saying that a query q is true in \mathcal{D} at any moment $\ell \in \mathbb{N}$, is defined as usual in temporal logic under the *strict semantics*; see, e.g., [28, 29] and further references therein:

$$\begin{aligned} \mathcal{D}, \ell \models \top, \quad \mathcal{D}, \ell \models A \text{ iff } A(\ell) \in \mathcal{D}, \quad \mathcal{D}, \ell \models q' \wedge q'' \text{ iff } \mathcal{D}, \ell \models q' \text{ and } \mathcal{D}, \ell \models q'', \\ \mathcal{D}, \ell \models \bigcirc q' \text{ iff } \mathcal{D}, \ell + 1 \models q', \quad \mathcal{D}, \ell \models \Diamond q' \text{ iff } \mathcal{D}, m \models q', \text{ for some } m > \ell. \end{aligned}$$

An *answer* to a query q over a data instance \mathcal{D} is any ℓ such that $0 \leq \ell \leq \max \mathcal{D}$ and $\mathcal{D}, \ell \models q$. The *temporal depth* $tdp(q)$ of q is the maximum number of nested temporal operators in q ; the *size* $|q|$ of q is the number of symbols in it.

Note that there is a close link between evaluating path queries and algorithms for finding patterns in strings [23]. A *sequence* is a data instance $\mathcal{D} = \delta_0 \dots \delta_m$ with $\delta_0 = \emptyset$ and $|\delta_i| = 1$, for $0 < i \leq m$. A *sequence query* is a path query of the form (1) with $\rho_0 = \emptyset$ and $|\rho_i| = 1$, for $0 < i \leq n$. We say that $\rho_1 \dots \rho_n$ is a *subsequence* of \mathcal{D} if there are $0 < i_1 < \dots < i_n \leq m$ with $\delta_{i_j} = \rho_j$, for $1 \leq j \leq n$, and we say that $\rho_1 \dots \rho_n$ is a *subword* of \mathcal{D} if there is $k \leq m$ with $\delta_{k+j} = \rho_j$, for $1 \leq j \leq n$. Querying sequences using sequence queries corresponds to the following matching problems:

- for any sequence query $q \in \mathcal{Q}_p[\Diamond]$ of the form (1), we have $\mathcal{D}, 0 \models q$ iff $\rho_1 \dots \rho_n$ is a subsequence of \mathcal{D} ;
- for any sequence query $q \in \mathcal{Q}_{in}$ of the form (1), we have $\mathcal{D}, 0 \models q$ iff $\rho_1 \dots \rho_n$ is a subword of \mathcal{D} .

We write $q \models q'$ if $\mathcal{D}, 0 \models q$ implies $\mathcal{D}, 0 \models q'$ for all \mathcal{D} , and $q \equiv q'$ if $q \models q'$ and $q' \models q$, in which case q and q' are called *equivalent*. For example, for any query q , we have $\Diamond \bigcirc q \equiv \bigcirc \Diamond q \equiv \Diamond \Diamond q \equiv \Diamond(\top \wedge \Diamond q)$. It follows that every $\mathcal{Q}_p[\bigcirc \Diamond]$ -query is equivalent to a query q of the form (1), in which $\rho_n \neq \top$ and whenever $\rho_i = \top$, $0 < i < n$, then $\rho_i = \rho_{i+1}$; in this case, we say that q is in *normal form*. Conversion to normal form can be done in L.

A query $q \in \mathcal{Q}[\Diamond]$ is in *normal form* if $q = \rho \wedge q_1 \wedge \dots \wedge q_n$, where ρ is a conjunction of atoms and each q_i , for $i = 1, \dots, n$, is a $\mathcal{Q}_p[\Diamond]$ -query (in normal form) that starts with \Diamond . Again, every $q \in \mathcal{Q}[\Diamond]$ can be converted to normal form in L using the equivalence

$$\rho_0 \wedge \Diamond(\rho_1 \wedge \bigwedge_{i=1}^n \Diamond q_i) \equiv \rho_0 \wedge \bigwedge_{i=1}^n \Diamond(\rho_1 \wedge \Diamond q_i).$$

For example, $\Diamond(A \wedge \Diamond B \wedge \Diamond C) \equiv \Diamond(A \wedge \Diamond B) \wedge \Diamond(A \wedge \Diamond C)$.

A query $q \in \mathcal{Q}[\bigcirc \Diamond]$ is in *normal form* if $q = r \wedge q_1 \wedge \dots \wedge q_n$, where r is a $\mathcal{Q}_p[\bigcirc]$ -query—that is, a \Diamond -free $\mathcal{Q}_p[\bigcirc, \Diamond]$ -query—and each q_i takes the form $\Diamond(r_{1,i} \wedge \Diamond(r_{2,i} \wedge \dots \wedge \Diamond(r_{n_i,i})))$ with $\mathcal{Q}_p[\bigcirc]$ -queries $r_{j,i}$, for $j = 1, \dots, n_i$. Each $q \in \mathcal{Q}[\bigcirc \Diamond]$ can be converted to normal form in polytime using the equivalence above, $\bigcirc \Diamond q \equiv \bigcirc \Diamond q$ and $\bigcirc(q \wedge q') \equiv (\bigcirc q \wedge \bigcirc q')$. For example, $\bigcirc(\bigcirc A \wedge \bigcirc(B \wedge \bigcirc C \wedge \bigcirc D)) \equiv \bigcirc^2 A \wedge \bigcirc(\bigcirc^2 B \wedge \bigcirc^3 C \wedge \bigcirc^2 D) \equiv \bigcirc^2 A \wedge \bigcirc(\bigcirc^2 B \wedge \bigcirc^3 C) \wedge \bigcirc(\bigcirc^2 B \wedge \bigcirc^2 D)$.

Our concern in this paper is the containment problem, which has become fundamental in database theory since the seminal work of Chandra and Merlin [24].

3. Complexity of Query Containment

The *query containment problem* for a class \mathcal{Q} of queries is formulated as follows: given any queries $q, q' \in \mathcal{Q}$, decide whether $q \models q'$.

Example 1. Consider the $\mathcal{Q}_p[\bigcirc \Diamond]$ -queries q_1, q_2 and q' below:

$$q_1 = \Diamond(A \wedge B \wedge \bigcirc A), \quad q_2 = \Diamond(A \wedge B \wedge \bigcirc B), \quad q' = \Diamond(B \wedge \Diamond(A \wedge B)).$$

It is readily seen that $q_1 \not\models q'$ (as the data instance $\emptyset\{A, B\}\{A\}$ makes q_1 true at 0 but not q') and $q_2 \not\models q'$. However, for the $\mathcal{Q}[\bigcirc \Diamond]$ -query $q_1 \wedge q_2$, we have $q_1 \wedge q_2 \models q'$. To show this, we observe that

any data instance $\mathcal{D} = \delta_0 \dots \delta_m$ satisfying $\mathbf{q}_1 \wedge \mathbf{q}_2$ at n has $\delta_i \supseteq \{A, B\}$, $\delta_{i+1} \supseteq \{A\}$, $\delta_j \supseteq \{A, B\}$, $\delta_{j+1} \supseteq \{B\}$, for some $n < i, j < m$. In each of the three cases, $i < j$, $i = j$, and $i > j$, we have $\mathcal{D}, n \models \mathbf{q}'$.

In contrast to conjunctive queries in first-order logic, where query containment is NP-complete [24], query containment is tractable for the majority of query classes defined above:

Theorem 1. *The query containment problems for $\mathcal{Q}_p[\circ\Diamond]$, $\mathcal{Q}[\Diamond]$ and their subclasses are all in L.*

Proof. Consider first $\mathcal{Q}_p[\circ\Diamond]$. Suppose we are given two queries $\mathbf{q}, \mathbf{q}' \in \mathcal{Q}_p[\circ\Diamond]$ in normal form, \mathbf{q} takes the form (1) and $\mathbf{q}' = \rho'_0 \wedge \mathbf{o}'_1(\rho'_1 \wedge \mathbf{o}'_2(\rho'_2 \wedge \dots \wedge \mathbf{o}'_m \rho'_m))$, where $\mathbf{o}'_i \in \{\circ, \Diamond\}$ and the ρ'_i are conjunctions of atoms (regarded as sets).

For any $k \in \mathbb{N}$, we denote by $[k]$ the closed interval $[0, k] \subseteq \mathbb{N}$. A function $h: [m] \rightarrow [n]$ is *monotone* if $h(i) < h(j)$ whenever $i < j$. A monotone function $h: [m] \rightarrow [n]$ is called a *containment witness* for \mathbf{q} and \mathbf{q}' if the following conditions hold:

- $h(0) = 0$;
- $\rho'_i \subseteq \rho_{h(i)}$, for all $i \in [m]$;
- if $i < m$ and $\mathbf{o}'_{i+1} = \circ$, then $\mathbf{o}_{h(i)+1} = \circ$ and $h(i+1) = h(i) + 1$.

Note that checking the existence of a containment witness can be done in L in $|\mathbf{q}|$ and $|\mathbf{q}'|$. Indeed, consider first the case of $\mathcal{Q}_p[\Diamond]$ -queries. It is easily seen that a containment witness for \mathbf{q} and \mathbf{q}' exists iff there is a sequence of pairs $(0, j_0), \dots, (m, j_n)$, where $j_0 = 0$ and j_{i+1} is the *minimal* number $j > j_i$ such that $\rho'_{i+1} \subseteq \rho_j$. For $\mathcal{Q}_p[\circ\Diamond]$ -queries, we need a more involved procedure that relies, however, on the same idea. Let $k'_0 \geq 0$ be the minimal index such that $\mathbf{o}'_{k'_0+1} = \Diamond$ (it follows that $\mathbf{o}'_1 = \dots = \mathbf{o}'_{k'_0} = \circ$). If no such k'_0 exists, we set $k'_0 = m$. We check if $\mathbf{o}_1 = \dots = \mathbf{o}_{k'_0} = \circ$ and $\rho'_0 \subseteq \rho_0, \dots, \rho'_{k'_0} \subseteq \rho_{k'_0}$. If this is not the case, a containment witness does not exist. If this is the case and $k'_0 = m$, a containment witness exists. Otherwise, let $k'_1 \geq 0$ be the minimal number such that $\mathbf{o}'_{k'_0+1+k'_1+1} = \Diamond$. If no such k'_1 exists, we chose k'_1 such that $k'_0 + 1 + k'_1 = m$. We find the *minimal* $k_0 > k'_0$ such that $\mathbf{o}_{k_0+1} = \dots = \mathbf{o}_{k_0+k'_1} = \circ$ and $\rho'_{k'_0+1} \subseteq \rho_{k_0}, \dots, \rho'_{k'_0+1+k'_1} \subseteq \rho_{k_0+k'_1}$. If such k_0 does not exist, a containment witness does not exist. If such k_0 exists and $k'_0 + 1 + k'_1 = m$, a containment witness exists. Otherwise, we proceed to find $k'_2 \geq 0$ and $k_1 > k_0$ that satisfy the conditions analogous to the above. We proceed until we either decide that a containment witness does not exist, or there is an iteration i when we choose $k'_i \geq 0$ such that $k'_0 + 1 + k'_1 + 1 + \dots + k'_{i-1} + 1 + k'_i = m$. Our procedure stops at this iteration.

The tractability of containment for path queries is an immediate consequence of the following analogue of the Chandra-Merlin criterion for conjunctive queries in first-order logic:

Lemma 1. *For any $\mathbf{q}, \mathbf{q}' \in \mathcal{Q}_p[\circ\Diamond]$ in normal form, we have $\mathbf{q} \models \mathbf{q}'$ iff there is a containment witness for \mathbf{q} and \mathbf{q}' .*

Proof. Given a data instance \mathcal{D} , by a *satisfying function* for \mathbf{q} of the form (1) in \mathcal{D} we mean any monotone function $f: [n] \rightarrow [\max \mathcal{D}]$ such that, for all $i \in [n]$, we have $f(0) = 0$, $\rho_i \subseteq \{A \mid A(f(i)) \in \mathcal{D}\}$, and if $\mathbf{o}_{i+1} = \circ$, then $f(i+1) = f(i) + 1$. It follows directly from the definition of the truth-relation that $\mathcal{D}, 0 \models \mathbf{q}$ iff there exists a satisfying function for \mathbf{q} in \mathcal{D} .

Suppose h is a containment witness for \mathbf{q} and \mathbf{q}' . Take any data instance \mathcal{D} with $\mathcal{D}, 0 \models \mathbf{q}$. Let f be a satisfying function for \mathbf{q} in \mathcal{D} . Then it is readily checked that the composition $hf: [m] \rightarrow [\max \mathcal{D}]$ is a satisfying function for \mathbf{q}' in \mathcal{D} , and so $\mathcal{D}, 0 \models \mathbf{q}'$. It follows that $\mathbf{q} \models \mathbf{q}'$.

Conversely, suppose $\mathbf{q} \models \mathbf{q}'$. We define a containment witness for \mathbf{q} and \mathbf{q}' by induction on the temporal depth $tdp(\mathbf{q}')$ of \mathbf{q}' . If $tdp(\mathbf{q}') = 0$, then $\mathbf{q}' = \rho'_0$ and $\rho'_0 \subseteq \rho$. It follows that $h(0) = 0$ is the required containment witness. As an induction hypothesis (IH), we assume next that there is a containment witness for any queries \mathbf{q}_1 and \mathbf{q}_2 such that $\mathbf{q}_1 \models \mathbf{q}_2$ and $tdp(\mathbf{q}_2) < tdp(\mathbf{q}')$. Two cases are possible.

Case 1: $\sigma'_1 = \circ$. Let $k > 0$ be the maximal index such that $\sigma'_1 = \dots = \sigma'_k = \circ$. Then $\rho'_k \neq \top$ (because q' is in normal form) and $\sigma'_{k+1} = \diamond$ if $m > k$. Let r be the minimal index such that

$$\rho_0 \wedge \sigma_1(\rho_1 \wedge \sigma_2(\rho_2 \wedge \dots \wedge \sigma_r \rho_r)) \models \rho'_0 \wedge \sigma'_1(\rho'_1 \wedge \sigma'_2(\rho'_2 \wedge \dots \wedge \sigma'_k \rho'_k)).$$

We claim that, for all $i \leq k$, we have $\sigma_i = \circ$, $\rho_i \supseteq \rho'_i$, and so $r = k$. Indeed, if there is $i \leq k$ with $\sigma_i = \diamond$, then we take the data instance $\mathcal{D} = \rho_0 \dots \rho_{i-1} \emptyset^m \rho_i \dots \rho_n$ and obtain $\mathcal{D}, 0 \models q$ and $\mathcal{D}, 0 \not\models q'$, contrary to $q \models q'$. And if there is $i \leq k$ with $\rho_i \not\supseteq \rho'_i$, then we take the data instance $\mathcal{D} = \rho_0 \dots \rho_n$ and again obtain $\mathcal{D}, 0 \models q$ and $\mathcal{D}, 0 \not\models q'$.

Next, we consider the ‘tails’

$$q_{k+1} = \sigma_{k+1}(\rho_{k+1} \wedge \dots \wedge \sigma_n \rho_n), \quad q'_{k+1} = \sigma'_{k+1}(\rho'_{k+1} \wedge \dots \wedge \sigma'_m \rho'_m)$$

and observe that $q \models q'$ implies $q_{k+1} \models q'_{k+1}$. As $\text{tdp}(q'_{k+1}) < \text{tdp}(q')$, the IH gives a containment witness h' for q_{k+1} and q'_{k+1} . Using it, we define the required containment witness h for q and q' by taking $h(i) = i$, for all $i \leq k$, and $h(k+j) = h'(j) + k$, for all j with $1 \leq j \leq m-k$.

Case 2: $\sigma'_1 = \diamond$. Suppose first that there is a minimal initial subquery

$$\bar{q}'_k = \rho'_0 \wedge \sigma'_1(\rho'_1 \wedge \dots \wedge \sigma'_k \rho'_k)$$

of q' such that $\rho'_k \neq \top$, $\sigma'_{k+1} = \diamond$ and $k > 0$. Let $\bar{q}_r = \rho_0 \wedge \sigma_1(\rho_1 \wedge \dots \wedge \sigma_r \rho_r)$ be the minimal initial subquery of q with $\bar{q}_r \models \bar{q}'_k$. As $\text{tdp}(\bar{q}'_k) < \text{tdp}(q')$, the IH gives a containment witness h_0 for \bar{q}_r and \bar{q}'_k . As $\rho'_k \neq \top$ and r is minimal, $h_0(k) = r$. By IH, our claim also holds for $q'_{k+1} = \sigma'_{k+1}(\rho'_{k+1} \wedge \dots \wedge \sigma'_m \rho'_m)$ and $q_{r+1} = \sigma_{r+1}(\rho_{r+1} \wedge \dots \wedge \sigma_n \rho_n)$ and gives a containment witness h_1 for q_{r+1} and q'_{k+1} . We then obtain a containment witness for q and q' by concatenating h_0 and h_1 .

Finally, suppose that there is no $\rho'_k \neq \top$ with $\sigma'_{k+1} = \diamond$ and $k > 0$. Then q' takes the form

$$\rho'_0 \wedge \diamond^k(\rho'_k \wedge \sigma'_{k+1} \rho'_{k+1} \dots \wedge \sigma'_m \rho'_m),$$

where $k > 0$, $\rho'_k \neq \top$, and $\sigma'_{k+1} = \dots = \sigma'_m = \circ$. We claim that there exists $j \geq k$ such that $\sigma_{j+1} = \dots = \sigma_{j+m-k} = \circ$ and $\rho_{j+\ell} \supseteq \rho'_{k+\ell}$, for all ℓ with $0 \leq \ell \leq m-k$, which clearly implies the existence of a containment witness for q and q' . To prove this claim, suppose there is no such j . Consider the data instance $\mathcal{D} = \rho_0 w_1 \rho_1 \dots w_n \rho_n$, where $w_i = \varepsilon$ (the empty word) if $\sigma_i = \circ$, and $w_i = \emptyset^m$ if $\sigma_i = \diamond$. Then $\mathcal{D}, 0 \models q$ but $\mathcal{D}, 0 \not\models q'$, contrary to $q \models q'$. \dashv

Now we prove Theorem 1 for the class $\mathcal{Q}[\diamond]$ of queries of the (normal) form $\rho \wedge q_1 \wedge \dots \wedge q_n$, where ρ is a conjunction of atoms and each q_i is a $\mathcal{Q}_p[\diamond]$ -query in normal form that starts with \diamond . The tractability of containment for $\mathcal{Q}[\diamond]$ -queries follows from Lemma 1 and the next criterion:

Lemma 2. *If $q = \rho \wedge q_1 \wedge \dots \wedge q_n \in \mathcal{Q}[\diamond]$ is in normal form and $q' \in \mathcal{Q}_p[\diamond]$, then $q \models q'$ iff $\rho \wedge q_i \models q'$, for some i , $1 \leq i \leq n$.*

To illustrate, consider the queries q_1 and q_2 from Example 1, in which we replace \circ by \diamond , and q' . It is easy to see that $q_1 \not\models q'$, $q_2 \not\models q'$, and so $q_1 \wedge q_2 \not\models q'$. To check the latter, consider $\mathcal{D} = \emptyset\{A, B\}\{A\}\{B\}$, which satisfies $A \wedge B$ from q_1 and q_2 at the same time instant, and then A from q_1 and B from q_2 at different time instants.

Proof. (\Rightarrow) Suppose that $q' = \rho^0 \wedge \diamond(\rho^1 \wedge \diamond(\rho^2 \wedge \dots \wedge \diamond \rho^m))$, $q_i = \diamond(\rho_i^1 \wedge \diamond(\rho_i^2 \wedge \dots \wedge \diamond \rho_i^{n_i}))$, for $1 \leq i \leq n$, and $q \models q'$. Note that $\rho \supseteq \rho^0$ as otherwise $q \not\models q'$. For each i , $1 \leq i \leq n$, we define inductively a function

$$f_i: \{1, \dots, m\} \rightarrow \{1, \dots, n_i\} \cup \{\infty\}.$$

To begin with, we set $f_i(1) = j$ if j is minimal such that $\rho_i^j \supseteq \rho^1$ and $f_i(1) = \infty$ if no j with $\rho_i^j \supseteq \rho^1$ exists. Further, inductively, if $f_i(\ell) = \infty$, then $f_i(\ell+1) = \infty$; if $f_i(\ell) < \infty$, then we set $f_i(\ell+1) = j$ if j is minimal such that $j > f_i(\ell)$ and $\rho_i^j \supseteq \rho^{\ell+1}$; if no $j > f_i(\ell)$ with $\rho_i^j \supseteq \rho^{\ell+1}$ exists, we set

$f_i(\ell + 1) = \infty$. It follows immediately from the definition that if there is $i \leq n$ such that $f_i(m) < \infty$, then $\rho \wedge \mathbf{q}_i \models \mathbf{q}'$, as required. So suppose there is no such $i \leq n$ and derive a contradiction by proving that in this case $\mathbf{q} \not\models \mathbf{q}'$.

Let $m' \leq m$ be minimal such that $f_i(m') = \infty$ for all $i \leq n$. Consider the data instance $\mathcal{D}_1 = \rho \rho_1^1 \dots \rho_1^{k_1} \dots \rho_n^1 \dots \rho_n^{k_n}$, where $k_i = \min\{n_i, f_i(1) - 1\}$, $1 \leq i \leq n$ (we set $\infty - 1 = \infty$). If $m' = 1$, then $\mathcal{D}_1, 0 \models \mathbf{q}$ and $\mathcal{D}_1, 0 \not\models \mathbf{q}'$ since $\mathcal{D}_1, 0 \not\models \diamond \rho^1$, and we are done. Otherwise, for $2 \leq \ell \leq m'$, we take

$$\delta_\ell = \bigcup_{1 \leq i \leq n, f_i(\ell) < \infty} \rho_i^{f_i(\ell)} \quad \text{and} \quad \mathcal{D}_\ell = \rho_1^{f_1(\ell)+1} \dots \rho_1^{k_{\ell,1}} \dots \rho_n^{f_n(\ell)+1} \dots \rho_n^{k_{\ell,n}},$$

where $k_{\ell,i} = \min\{n_i, f_i(\ell + 1) - 1\}$, for $1 \leq i \leq n$ (note that $\rho_i^{f_i(\ell)+1} \dots \rho_i^{k_{\ell,i}}$ is empty if $f_i(\ell) + 1 > n_i$). Then we set

$$\mathcal{D} = \mathcal{D}_1 \delta_1 \mathcal{D}_2 \delta_2 \dots \delta_{m'-1} \mathcal{D}_{m'}.$$

It follows from the construction that $\mathcal{D}, 0 \models \mathbf{q}$ and $\mathcal{D}, 0 \not\models \mathbf{q}'$, contrary to $\mathbf{q} \models \mathbf{q}'$.

The implication (\Leftarrow) is obvious. ←

This completes the proof of Theorem 1. ←

For queries $\mathbf{q} \in \mathcal{Q}[\diamond]$, Lemma 2 does not hold as illustrated by Example 1. Moreover, in contrast to Theorem 1, we have the following:

Theorem 2. *The query containment problem for $\mathcal{Q}[\diamond]$ is coNP-complete.*

Proof. To show the upper bound, suppose we are given two queries $\mathbf{q}, \mathbf{q}' \in \mathcal{Q}[\diamond]$ in normal form, where

$$\begin{aligned} \mathbf{q} &= \mathbf{r} \wedge \mathbf{q}_1 \wedge \dots \wedge \mathbf{q}_n \quad \text{with} \quad \mathbf{q}_i = \diamond(\mathbf{r}_{i,1} \wedge \diamond(\mathbf{r}_{i,2} \wedge \dots \wedge \diamond \mathbf{r}_{i,n_i})), \\ &\quad \mathbf{r}, \mathbf{r}_{i,j} \in \mathcal{Q}_p[\diamond], \quad \text{for } i = 1, \dots, n, j = 1, \dots, n_i, \\ \mathbf{q}' &= \mathbf{r}' \wedge \mathbf{q}'_1 \wedge \dots \wedge \mathbf{q}'_{n'} \quad \text{with} \quad \mathbf{q}'_i = \diamond(\mathbf{r}'_{i,1} \wedge \diamond(\mathbf{r}'_{i,2} \wedge \dots \wedge \diamond \mathbf{r}'_{i,n'_i})), \\ &\quad \mathbf{r}', \mathbf{r}'_{i,j} \in \mathcal{Q}_p[\diamond], \quad \text{for } i = 1, \dots, n', j = 1, \dots, n'_i. \end{aligned}$$

By definition, $\mathbf{q} \not\models \mathbf{q}'$ iff there exist a data instance \mathcal{D} and some m , $1 \leq m \leq n'$, such that $\mathcal{D} \models \mathbf{q}$ and $\mathcal{D} \not\models \mathbf{r}' \wedge \mathbf{q}'_m$. Our aim is to show that it suffices to consider \mathcal{D} with $\max \mathcal{D} \leq O(|\mathbf{q}| |\mathbf{q}'_m|)$. If this is the case, then an obvious NP-algorithm deciding $\mathbf{q} \not\models \mathbf{q}'$ would be to guess such m and \mathcal{D} with $\text{sig}(\mathcal{D}) = \text{sig}(\mathbf{q}) \cup \text{sig}(\mathbf{q}')$, and then check in polytime whether $\mathcal{D} \models \mathbf{q}$ and $\mathcal{D} \not\models \mathbf{r}' \wedge \mathbf{q}'_m$.

So, suppose we have $\mathcal{D} \models \mathbf{q}$ and $\mathcal{D} \not\models \mathbf{r}' \wedge \mathbf{q}'_m$, for some \mathcal{D} and m . As $\mathcal{D} \models \mathbf{q}$, for each \mathbf{q}_i , there is a *satisfying function* f_i in \mathcal{D} , which is a monotone function $f_i: [1, n_i] \rightarrow [\max \mathcal{D}]$ such that, for all $j \in [1, n_i]$, we have $\mathcal{D}_{f_i(j)} \mathcal{D}_{f_i(j)+1} \dots \mathcal{D}_{f_i(j)+\text{tdp}(\mathbf{r}_{i,j})} \models \mathbf{r}_{i,j}$. Let $R_{i,j} = [f_i(j), f_i(j) + \text{tdp}(\mathbf{r}_{i,j})]$, for $i \in [1, n]$ and $j \in [1, n_i]$. We may clearly assume that $\mathcal{D}_l = \emptyset$, for all $l \in [\max \mathcal{D}]$ with $l \notin \bigcup_{i=1}^n \bigcup_{j=1}^{n_i} R_{i,j} \cup [\text{tdp}(\mathbf{r})]$. Now, we cut certain segments from \mathcal{D} maintaining the property that the resulting data instance \mathcal{D}' makes \mathbf{q} true and $\mathbf{r}' \wedge \mathbf{q}'_m$ false at time 0.

Suppose there exist $\mathbf{r}_{i,j}$ and $\mathbf{r}'_{i',j'}$ such that $f_{i'}(j') - (f_i(j) + \text{tdp}(\mathbf{r}_{i,j})) > \text{tdp}(\mathbf{q}'_m)$ and $(f_i(j) + \text{tdp}(\mathbf{r}_{i,j}), f_{i'}(j')) \cap R_{k,l} = \emptyset$, for all $k \in [1, n]$ and $l \in [1, n_i]$. Then we remove from the segment $\mathcal{D}_{f_i(j)+\text{tdp}(\mathbf{r}_{i,j})+1} \dots \mathcal{D}_{f_{i'}(j')-1}$ of \mathcal{D} all \mathcal{D}_l with $l > f_i(j) + \text{tdp}(\mathbf{r}_{i,j}) + \text{tdp}(\mathbf{q}'_m) + 1$. By definition, the removed \mathcal{D}_l are all empty. The resulting shorter instance \mathcal{D}' is such that $\mathcal{D}' \models \mathbf{q}$ and $\mathcal{D}' \not\models \mathbf{r}' \wedge \mathbf{q}'_m$. Indeed, we have kept all the witnesses that make $\mathcal{D}' \models \mathbf{q}$ intact. Now, suppose $\mathcal{D}' \models \mathbf{r}' \wedge \mathbf{q}'_m$. We take the satisfying function f' for \mathbf{q}'_m in \mathcal{D}' and modify it to construct f'' such that $f''(j) = f'(j)$, for all $j \in [1, n'_m]$ with $f'(j) \leq f_i(j) + \text{tdp}(\mathbf{r}_{i,j})$, and $f''(j) = f'(j) + \ell$, for all j with $f'(j) > f_i(j) + \text{tdp}(\mathbf{r}_{i,j})$, where ℓ is the number of the \mathcal{D}_l that were removed from the segment $\mathcal{D}_{f_i(j)+\text{tdp}(\mathbf{r}_{i,j})+1} \dots \mathcal{D}_{f_{i'}(j')-1}$. It is readily seen that f'' is a satisfying function for \mathbf{q}'_m in \mathcal{D} , which is a contradiction. Thus, $\mathcal{D}' \not\models \mathbf{r}' \wedge \mathbf{q}'_m$.

By performing this operation for all suitable $\mathbf{r}_{i,j}$ and $\mathbf{r}_{i',j'}$, we obtain a data instance with at most

$$tdp(\mathbf{r}) + 1 + \sum_{1 \leq i \leq n, 1 \leq j \leq n_i} (tdp(\mathbf{r}_{i,j}) + 1) + N \sum_{1 \leq j \leq m'} (tdp(\mathbf{r}_{m',j}) + 1)$$

time instants, where N is the number of $\mathbf{r}_{i,j}$ in \mathbf{q} plus 1.

The matching lower bound is shown by reduction of the 3SAT problem to the complement of the containment problem for $\mathcal{Q}[\square\diamond]$. Suppose we are given a 3CNF $\varphi = c_1 \wedge \dots \wedge c_n$ with clauses c_i and variables x_1, \dots, x_m such that no c_i contains both x_j and $\neg x_j$, for any j . Our aim is to construct $\mathcal{Q}[\square]$ -queries \mathbf{r}_i , for all $i = 0, \dots, n$, and a $\mathcal{Q}[\square\diamond]$ -query \mathbf{r}' such that

$$\varphi \text{ is satisfiable} \quad \text{iff} \quad \bigwedge_{0 \leq i \leq n} \diamond \mathbf{r}_i \not\models \diamond \mathbf{r}'. \quad (2)$$

For each $j = 1, \dots, m$, we take two atoms X_j and \bar{X}_j to represent x_j and $\neg x_j$, respectively. Given a literal $\ell_j \in \{x_j, \neg x_j\}$, set $L_{\ell_j} = X_j$ if $\ell_j = x_j$ and $L_{\ell_j} = \bar{X}_j$ if $\ell_j = \neg x_j$. We also use two additional atoms B and E . We require the following conjunctions of atoms, written as sets:

$$\begin{aligned} \alpha &= \{X_1, \bar{X}_1, \dots, X_m, \bar{X}_m\}, \\ \alpha_B &= \alpha \cup \{B\}, \\ \lambda_\ell &= \alpha \setminus \{L_\ell\}, \text{ for a literal } \ell \text{ over } x_1, \dots, x_m, \\ \beta_j &= \alpha \setminus \{X_j, \bar{X}_j\}, \text{ for } j = 1, \dots, m. \end{aligned}$$

Let $\sigma = \{X_1, \bar{X}_1, \dots, X_m, \bar{X}_m, B, E\}$. We define the $\mathcal{Q}[\square]$ -queries \mathbf{r}_i as words of the form $\rho_0 \rho_1 \rho_2 \dots \rho_l$ over the alphabet 2^σ that represent $\rho_0 \wedge \square(\rho_1 \wedge \square(\rho_2 \wedge \dots \wedge \square \rho_l))$. Namely, we set

$$\mathbf{r}_0 = \{B\} \emptyset^{2m-1} \alpha \beta_1 \dots \beta_m \emptyset^{2m} \{E\}, \quad \text{and} \quad \mathbf{r}_i = \alpha_B \mathbf{r}_{i,1} \mathbf{r}_{i,2} \mathbf{r}_{i,3} \{E\}, \quad \text{for } i = 1, \dots, n,$$

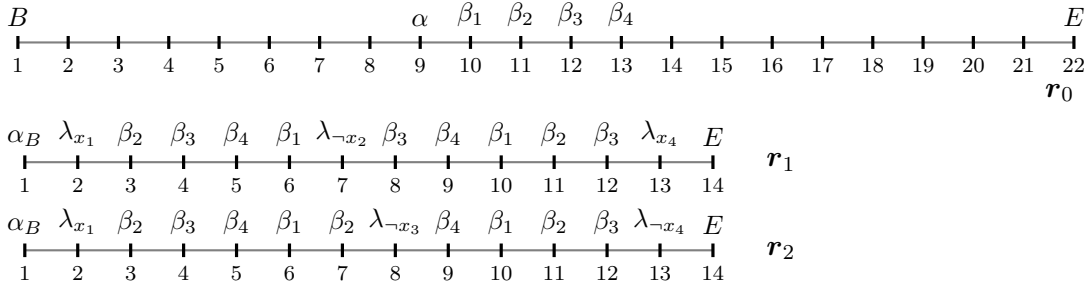
where the substrings $\mathbf{r}_{i,k}$, for $k = 1, 2, 3$, of \mathbf{r}_i are defined as follows: if $c_i = \ell_{j_1} \vee \ell_{j_2} \vee \ell_{j_3}$, then

$$\mathbf{r}_{i,k} = \beta_1 \dots \beta_{j_k-1} \lambda_{\ell_{j_k}} \beta_{j_k+1} \dots \beta_m.$$

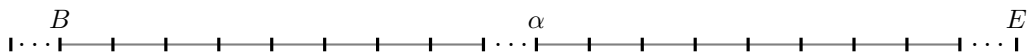
Thus, the length of each word \mathbf{r}_i , for $i = 1, \dots, m$, is $3m + 2$, and so the temporal depth of the corresponding queries \mathbf{r}_i is $3m + 1$. The length of \mathbf{r}_0 is $5m + 2$. Finally, we set

$$\mathbf{r}' = B \wedge \square^{2m} \diamond (\alpha \wedge \square^{2m} \diamond E).$$

Example 2. Consider the 3CNF $\varphi = c_1 \wedge c_2$ with $c_1 = x_1 \vee \neg x_2 \vee x_4$, $c_2 = x_1 \vee \neg x_3 \vee \neg x_4$, $n = 2$ and $m = 4$. The words $\mathbf{r}_0, \mathbf{r}_1, \mathbf{r}_2$ for φ are illustrated in the picture below, where the numbers indicate the positions of the respective characters, starting from 1, and \emptyset is omitted (remember that, in (2), we use the queries $\diamond \mathbf{r}_i$).



The query $\diamond \mathbf{r}'$ can be depicted as follows, with the dots \dots mimicking the \diamond -operators:



We have shown that the containment problem for the classes $\mathcal{Q}_p[\Box\Diamond]$ and $\mathcal{Q}[\Diamond]$ of *LTL*-queries lies in the complexity class L. It would be of interest to understand if this complexity bound is tight or the problem is easier, e.g., in NC¹. As observed earlier, unlike first-order conjunctive queries but similarly to XPath-queries and queries with transitive roles, $\mathcal{Q}[\Box\Diamond]$ -query containment is not polytime reducible to query evaluation (unless P = NP). However, it follows from the proofs above that, for $\mathcal{Q}_p[\Box\Diamond]$ and $\mathcal{Q}[\Diamond]$, containment is polytime reducible to evaluation, although the reduction is less trivial than in the first-order case. It is also worth noting that a polysize witness for non-containment exists for all of our queries, similarly to some classes of XPath/transitive queries [25].

In this paper, we have not considered conjunctive queries with the until-operator U , for which containment is only known to be in PSPACE. Natural restricted fragments of conjunctive path-queries with U that only allow conjunctions of atoms on the left-hand side of U have been identified in [20, 19]; however, the complexity of containment for those fragments have not been studied yet. For path- U -queries satisfying the restriction above, the existence of a polysize witness for non-containment was shown in [19, Theorem 9]. This fact implies a coNP upper bound for containment, but it is not known whether this bound is tight.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

- [1] H. Ono, A. Nakamura, On the size of refutation Kripke models for some linear modal and tense logics, *Studia Logica* (1980) 325–333.
- [2] A. P. Sistla, E. M. Clarke, The complexity of propositional linear temporal logics, *J. ACM* 32 (1985) 733–749. URL: <https://doi.org/10.1145/3828.3837>. doi:10.1145/3828.3837.
- [3] C.-C. Chen, I.-P. Lin, The computational complexity of the satisfiability of modal Horn clauses for modal propositional logics, *Theor. Comp. Sci.* 129 (1994) 95–121.
- [4] S. Demri, P. Schnoebelen, The complexity of propositional linear temporal logics in simple cases, *Information and Computation* 174 (2002) 84–103. URL: <https://www.sciencedirect.com/science/article/pii/S0890540101930949>. doi:<https://doi.org/10.1006/inco.2001.3094>.
- [5] M. Bauland, T. Schneider, H. Schnoor, I. Schnoor, H. Vollmer, The complexity of generalized satisfiability for linear temporal logic, in: H. Seidl (Ed.), *Foundations of Software Science and Computational Structures*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 48–62.
- [6] C. Dixon, M. Fisher, B. Konev, Tractable temporal reasoning, in: *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI’07*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007, p. 318–323.
- [7] A. Artale, R. Kontchakov, V. Ryzhikov, M. Zakharyashev, The complexity of clausal fragments of LTL, in: K. L. McMillan, A. Middeldorp, A. Voronkov (Eds.), *Logic for Programming, Artificial Intelligence, and Reasoning - 19th International Conference, LPAR-19*, Stellenbosch, South Africa, December 14–19, 2013. *Proceedings, volume 8312 of Lecture Notes in Computer Science*, Springer, 2013, pp. 35–52. URL: https://doi.org/10.1007/978-3-642-45221-5_3. doi:10.1007/978-3-642-45221-5_3.
- [8] A. Artale, R. Kontchakov, V. Ryzhikov, M. Zakharyashev, A cookbook for temporal conceptual data modelling with description logics, *ACM Trans. Comput. Log.* 15 (2014) 25:1–25:50. URL: <https://doi.org/10.1145/2629565>. doi:10.1145/2629565.
- [9] V. Fionda, G. Greco, LTL on finite and process traces: Complexity results and a practical reasoner, *J. Artif. Intell. Res.* 63 (2018) 557–623. URL: <https://doi.org/10.1613/jair.1.11256>. doi:10.1613/JAIR.1.11256.
- [10] A. Artale, R. Kontchakov, A. Kovtunova, V. Ryzhikov, F. Wolter, M. Zakharyashev, First-order rewritability of ontology-mediated queries in linear temporal logic, *Artif. Intell.* 299 (2021) 103536. URL: <https://doi.org/10.1016/j.artint.2021.103536>. doi:10.1016/j.artint.2021.103536.
- [11] J. Chomicki, D. Toman, M. H. Böhlen, Querying ATSQL databases with temporal logic, *ACM Trans. Database Syst.* 26 (2001) 145–178. URL: <https://doi.org/10.1145/383891.383892>. doi:10.1145/383891.383892.
- [12] J. Chomicki, D. Toman, Temporal logic in database query languages, in: L. Liu, M. T. Özsu (Eds.), *Encyclopedia of Database Systems*, Second Edition, Springer, 2018. URL: https://doi.org/10.1007/978-1-4614-8265-9_402. doi:10.1007/978-1-4614-8265-9_402.

- [13] A. Artale, R. Kontchakov, A. Kovtunova, V. Ryzhikov, F. Wolter, M. Zakharyashev, Ontology-mediated query answering over temporal data: A survey (invited talk), in: S. Schewe, T. Schneider, J. Wijsen (Eds.), 24th International Symposium on Temporal Representation and Reasoning, TIME 2017, October 16–18, 2017, Mons, Belgium, volume 90 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, pp. 1:1–1:37. URL: <https://doi.org/10.4230/LIPIcs.TIME.2017.1>. doi:10.4230/LIPIcs.TIME.2017.1.
- [14] S. Brandt, E. G. Kalayci, V. Ryzhikov, G. Xiao, M. Zakharyashev, Querying log data with metric temporal logic, *J. Artif. Intell. Res.* 62 (2018) 829–877. URL: <https://doi.org/10.1613/jair.1.11229>. doi:10.1613/jair.1.11229.
- [15] D. Wang, P. Hu, P. A. Walega, B. C. Grau, Meteor: Practical reasoning in datalog with metric temporal operators, in: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022, AAAI Press, 2022, pp. 5906–5913. URL: <https://doi.org/10.1609/aaai.v36i5.20535>. doi:10.1609/AAAI.V36I5.20535.
- [16] A. Kurucz, V. Ryzhikov, Y. Savateev, M. Zakharyashev, Deciding fo-rewritability of regular languages and ontology-mediated queries in linear temporal logic, *J. Artif. Intell. Res.* 76 (2023) 645–703. URL: <https://doi.org/10.1613/jair.1.14061>. doi:10.1613/JAIR.1.14061.
- [17] D. Neider, R. Roy, What Is Formal Verification Without Specifications? A Survey on Mining LTL Specifications, Springer Nature Switzerland, Cham, 2025, pp. 109–125. URL: https://doi.org/10.1007/978-3-031-75778-5_6. doi:10.1007/978-3-031-75778-5_6.
- [18] R. Raha, R. Roy, N. Fijalkow, D. Neider, Scalable anytime algorithms for learning fragments of linear temporal logic, in: D. Fisman, G. Rosu (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, Springer International Publishing, Cham, 2022, pp. 263–280.
- [19] M. Fortin, B. Konev, V. Ryzhikov, Y. Savateev, F. Wolter, M. Zakharyashev, Unique characterisability and learnability of temporal instance queries, in: G. Kern-Isberner, G. Lakemeyer, T. Meyer (Eds.), Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022, Haifa, Israel. July 31 - August 5, 2022, 2022. URL: <https://proceedings.kr.org/2022/17/>.
- [20] M. Fortin, B. Konev, V. Ryzhikov, Y. Savateev, F. Wolter, M. Zakharyashev, Reverse engineering of temporal queries mediated by LTL ontologies, in: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th–25th August 2023, Macao, SAR, China, ijcai.org, 2023, pp. 3230–3238. URL: <https://doi.org/10.24963/ijcai.2023/360>. doi:10.24963/IJCAI.2023/360.
- [21] J. C. Jung, V. Ryzhikov, F. Wolter, M. Zakharyashev, Extremal separation problems for temporal instance queries, in: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3–9, 2024, ijcai.org, 2024, pp. 3448–3456. URL: <https://www.ijcai.org/proceedings/2024/382>.
- [22] C. Fraser, Consistent subsequences and supersequences, *Theor. Comput. Sci.* 165 (1996) 233–246. URL: [https://doi.org/10.1016/0304-3975\(95\)00138-7](https://doi.org/10.1016/0304-3975(95)00138-7). doi:10.1016/0304-3975(95)00138-7.
- [23] M. Crochemore, C. Hancart, T. Lecroq, Algorithms on strings, Cambridge University Press, 2007.
- [24] A. Chandra, P. Merlin, Optimal implementation of conjunctive queries in relational data bases, in: Conference Record of the Ninth Annual ACM Symposium on Theory of Computing, 2–4 May 1977, Boulder, Colorado, USA, ACM, 1977, pp. 77–90.
- [25] G. Miklau, D. Suciu, Containment and equivalence for an xpath fragment, in: L. Popa, S. Abiteboul, P. G. Kolaitis (Eds.), Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3–5, Madison, Wisconsin, USA, ACM, 2002, pp. 65–76. URL: <https://doi.org/10.1145/543613.543623>. doi:10.1145/543613.543623.
- [26] H. Björklund, W. Martens, T. Schwentick, Conjunctive query containment over trees, *J. Comput. Syst. Sci.* 77 (2011) 450–472. URL: <https://doi.org/10.1016/j.jcss.2010.04.005>. doi:10.1016/J.JCSS.2010.04.005.
- [27] C. Haase, C. Lutz, Complexity of subsumption in the el family of description logics: Acyclic and

cyclic TBoxes, in: M. Ghallab, C. D. Spyropoulos, N. Fakotakis, N. M. Avouris (Eds.), ECAI 2008 - 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25, 2008, Proceedings, volume 178 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2008, pp. 25–29. URL: <https://doi.org/10.3233/978-1-58603-891-5-25>. doi:10.3233/978-1-58603-891-5-25.

- [28] D. Gabbay, A. Kurucz, F. Wolter, M. Zakharyashev, Many-Dimensional Modal Logics: Theory and Applications, volume 148 of *Studies in Logic*, Elsevier, 2003.
- [29] S. Demri, V. Goranko, M. Lange, Temporal Logics in Computer Science, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 2016.