Ontology-Based Query Answering for Probabilistic Temporal Data (Extended Version)

Patrick Koopmann

LTCS-Report 18-13

This is an extended version of the article to appear in the proceedings of AAAI 2019.
# Contents

1 Introduction ............................................. 2  
2 Preliminaries ............................................. 4  
3 Temporal Probabilistic Knowledge Bases and Queries ..................... 5  
4 Hardness of TPQ Entailment .................................. 8  
5 Deciding TPQ Entailment .................................... 9  
6 Positive TPQs ............................................. 11  
7 Conclusion ............................................... 12  
8 Acknowledgements .......................................... 12  
A Proofs of Theorems and Lemmata ................................... 14  
A.1 Lower Bounds ........................................... 15  
A.2 Correctness of the Decision Procedure ............................. 17  
A.3 Deciding r-Satisfiability .................................... 20  
A.4 Complexity Upper Bounds .................................. 25  
A.5 Positive Temporal Probabilistic Queries ........................... 26
Ontology-Mediated Query Answering for Probabilistic Temporal Data (Extended Version)*

Patrick Koopmann

November 14, 2018

Abstract

We investigate ontology-based query answering for data that are both temporal and probabilistic, which might occur in contexts such as stream reasoning or situation recognition with uncertain data. We present a framework that allows to represent temporal probabilistic data, and introduce a query language with which complex temporal and probabilistic patterns can be described. Specifically, this language combines conjunctive queries with operators from linear time logic as well as probability operators. We analyse the complexities of evaluating queries in this language in various settings. While in some cases, combining the temporal and the probabilistic dimension in such a way comes at the cost of increased complexity, we also determine cases for which this increase can be avoided.

1 Introduction

The internet has become highly dynamic, with information being frequently added and changed, and new data being generated from a variety of sources. In addition, new technologies such as smart phones and the internet of things (IoT) frequently encounter a data environment that is constantly changing. To make use of these data, there has been an increasing interest in investigating semantic and reasoning techniques that process not only static data, but streams of data, such as in the semantic stream reasoning paradigm [29]. One application is that of situation recognition, where we want to recognise or query temporal patterns in a stream of data. As [29] illustrate, frequently, the data encountered in stream reasoning applications is not only temporal, but also probabilistic in nature. In ontology-based query answering (OBQA), queries are evaluated with respect to an ontology, which specifies background knowledge about the domain of interest. Using a reasoner, this allows to query also information that follows implicitly from the data. While OBQA was originally designed for querying static and precise data, there is good motivation also for semantic stream reasoning as well as for querying historical data, where data are temporal and probabilistic.

As an example, consider a health or fitness monitoring application, for which one may want to use concepts from a medical ontology such as SNOMED CT [18] to describe information about the health status of a patient. Specifically, such an application could be used on a smartphone in combination with a sensor that measures the diastolic blood pressure of the patient while he is exercising [25]. As the sensor might be imprecise in its measurements, it might report information about whether the blood pressure of the patient is high with an associated probability, and provide this information to the application in regular time intervals. If a too high blood pressure was observed for several times during a short period, the application should give a warning to the patient, and advise him to take a break from his exercise.

*Supported by the DFG within the collaborative research center SFB 912 (HAEC).
We assume a representation of the data in form of a sequence of probabilistic data sets, which may have been obtained using further preprocessing and windowing operations. A typical query would then ask whether, with a high probability, the patient had at least twice a high blood pressure during the last 10 minutes. In order to properly take both the temporal and the probabilistic aspects of this question into account when querying the probabilistic stream, we propose a query language for OBQA that comes with both temporal and probabilistic operators. In this language, the query would be expressed as follows, where \textit{HighBloodPressure} is a concept defined in the ontology.

\[
P_{>8}(\bigcirc^{10}\Diamond(\text{HighBloodPressure}(x))
\wedge \bigcirc\Diamond(\text{HighBloodPressure}(x)))
\]

Our language is an extension of the well-investigated temporal query language introduced in [8, 3], which extends conjunctive queries with operators from linear temporal logic (LTL). Other authors considered using these operators also as part of the DL, either to describe temporal concepts [20], or to make the axioms of the ontology itself temporal [4]. Recently, this work has been extended also to metric temporal logics [2, 10]. A major restriction of using temporal concepts in the DL is that we cannot keep relations between objects stable throughout the timeline (rigid) without making the DL undecidable. This limits their application for querying situations concerning more than one object, which in applications more involved than in our running example might be crucial. For this reason, we focus on extensions of the query language rather than the DL in this paper. Temporal reasoning for streams of data has also been considered in the context of Datalog [31]. A recent survey on temporal query answering with ontologies can be found in [1].

However, all this work assumes the data to be certain, and thus cannot handle the probabilistic scenario described above. An OBQA framework for \textit{probabilistic} data was presented in [24], though a temporal dimension was not considered here yet. Since this publication, several authors investigated OBQA in similar settings [6, 5, 15]. In addition to settings based on probabilistic databases, there is also research on extending DLs with probability operators, such as in P\textit{-SHIF(D)}/P\textit{-SHOIN(D)} [26] or Prob\textit{-ALC}/Prob\textit{-EL} [23]. The probability operator used in our query language syntactically and semantically corresponds to the probability operator in Prob\textit{-ALC} and Prob\textit{-EL}.

To our knowledge, the only work that combines both temporal and probabilistic query answering in the presence of description logic ontologies is [14]. Albeit, the authors consider a different setting, in which
the flow of time is modeled by a Markov-process. [17] consider temporal probabilistic databases with temporal Datalog rules and constraints, and computing probabilities of conjunctive queries in these KBs. Both works do not allow for nested probabilities as part of the query language.

To handle scenarios like in our example, we propose a framework that combines the ideas from [8] for temporal knowledge bases with the framework for probabilistic knowledge bases introduced in [24]. To query data in the resulting temporal probabilistic knowledge bases, our language extends temporal queries with probabilistic operators, to allow to assign probability bounds to arbitrary parts of the query. We establish a more or less complete picture of the complexity of query entailment in this framework for various DLs (see Figure 1 explained in detail throughout the text), and also discuss a restricted variant of our query language without negation, which sometimes leads to a restricted complexity.

Detailed proofs can be found in the appendix.

2 Preliminaries

We recall the DLs studied in the paper, conjunctive query answering, and probabilistic complexity classes.

Description Logics. Let $N_C$, $N_R$ and $N_l$ be pair-wise countably infinite sets of respectively concept names, role names and individual names. A role is an expression of the forms $r$, $r^−$, where $r ∈ N_R$. Concepts are of the following forms, where $A ∈ N_C$, $R$, $C$ are concepts, $n ∈ N$ and $a ∈ N_l$:

$$A | C \cap D | ∃R.C | ∀R.C | ≥ nR.C | \{a\}.$$  

A TBox is a set of axioms of the forms $C ⊆ D$, $R ⊆ S$ and $\text{trans}(R)$, where $C,D$ are concepts and $R, S$ roles, while an ABox is a set of assertions of the forms $A(a)$ and $r(a, b)$, $A ∈ N_C$, $r ∈ N_R$, $a, b ∈ N_l$. For a TBox $T$, we define the relation $\prec_T$ s.t. for two roles $R, S$, $S \prec_T R$ holds if $S' ⊆ R' ∈ T$ with $S, S' ∈ \{s, s^-\}$ and $R, R' ∈ \{r, r^-\}$, $r, s ∈ N_R$. A role $R$ is complex wrt. $T$ if $\text{trans}(S) ∈ T$ for some role $S$ s.t. $S \prec^+_T R$. To ensure decidability, we require for every concept of the form $≥ nR.C$ in $T$ that $R$ is not complex. Now a knowledge base (KB) is a tuple $(T, A)$ of a TBox $T$ and an ABox $A$. We differentiate different DLs based on the operators allowed: $\mathcal{EL}$ only supports concepts of the form $A$, $C \cap D$ and $∃R.C$ and axioms of the form $C ⊆ D$, no roles of the form $r^-$, and no axioms of the forms $R ⊆ S$ or $\text{trans}(R)$. $\mathcal{ALC}$ extends $\mathcal{EL}$ with concepts of the form $¬C$, and $\mathcal{S}$ extends $\mathcal{ALC}$ with axioms of the form $\text{trans}(R)$. More expressive DLs are denoted by attaching a letter to the DL, where we use $\mathcal{I}$ for support of roles $r^-$, $\mathcal{O}$ for concepts of the form $\{a\}$, $\mathcal{Q}$ for concepts of the form $≥ nR.C$, and $\mathcal{H}$ for axioms of the form $R ⊆ S$. For example, $\mathcal{SH\\!}\mathcal{\mathcal{T}}$ extends $\mathcal{S}$ with axioms of the form $R ⊆ S$ and roles of the form $r^-$, whereas $\mathcal{ALCD\\!}\mathcal{H}$ extends $\mathcal{ALC}$ with concepts of the form $\{a\}$ and $≥ nR.C$. Depending on the DL $\mathcal{L}$ used, we speak of $\mathcal{L}$ concepts, $\mathcal{L}$ axioms, $\mathcal{L}$ TBoxes and $\mathcal{L}$ KBs.

The semantics of KBs is defined in terms of interpretations $I = (\Delta^\mathcal{L}, \cdot^I)$, where $\Delta^\mathcal{L}$ is a set of domain elements and $\cdot^I$ maps each concept name $A ∈ N_C$ to a set $A^I ⊆ \Delta^\mathcal{L}$, each role name $r ∈ N_R$ to a relation $r^I ⊆ \Delta^\mathcal{L} × \Delta^\mathcal{L}$, each individual name $a ∈ N_l$ to a domain element $a^I ∈ \Delta^\mathcal{L}$, and each role $r^-$ to $(r^-)^I = (r^I)^-$. It is extended to concepts by

$$(C \cap D)^I = C^I \cap D^I,$$$$(¬C)^I = \Delta^\mathcal{L} \setminus C^I,$$$$(∃R.C)^I = \{d ∈ I | \exists e ∈ \Delta^\mathcal{L} : (d, e) ∈ R^I, e ∈ C^I\},$$

$$(≥ nR.C)^I = \{d ∈ I | \#\{e ∈ C^I | (d, e) ∈ R^I, e ∈ C^I\} ≥ n\}.$$  

We say that an interpretation $I$ satisfies an axiom/assertion $α$, in symbols $I ∶|= α$, if $α = C \subseteq D$ and $C^I ⊆ \Delta^\mathcal{L}$; $α = R ⊆ S$ and $R^I ⊆ S^I$; $α = \text{trans}(R)$ and $R^I = (R^I)^+$; $α = A(a)$ and $a^I ∈ A^I$; and $α = r(a, b)$ and $(a^I, b^I) ∈ r^I$. $I$ is a model of a TBox/ABox/KB iff it satisfies all axioms in it. Finally, a TBox/ABox/KB $\mathcal{X}$ entails an axiom/assertion $α$ iff $α$ is satisfied by every model of $\mathcal{X}$.
Conjunctive Queries. A conjunctive query (CQ) takes the form $q = \exists \bar{x}. \phi(\bar{x}, \bar{y})$, where $\bar{x}, \bar{y}$ are vectors of variables and $\phi(\bar{x}, \bar{y})$ is a conjunction over atoms of the forms $A(t_1)$ and $r(t_1, t_2)$, where $A \in \mathbb{N}_C$ and $r \in \mathbb{N}_R$ is not complex, and $t_1$ and $t_2$ are terms taken from $\mathbb{N}_I$. $\bar{x}$ and $\bar{y}$ are the answer variables of $q$. Given an interpretation $\mathcal{I}$ and a CQ $q$ with answer variables $x_1, \ldots, x_n$, the vector $a_1 \ldots a_n \subseteq \mathbb{N}_I^n$ is an answer of $q$ in $\mathcal{I}$ if there exists a mapping $\pi: \text{term}(q) \rightarrow \Delta^2$ s.t. $\pi(x_i) = a_i$ for $i \in [1, n]$, $\pi(b) = b^2$ for $b \in \mathbb{N}_I$, $\pi(t) \in A^2$ for every $A(t)$ in $q$, and $(\pi(t_1), \pi(t_2)) \in r^2$ for every $r(t_1, t_2)$ in $q$. A vector $a_1 \ldots a_n$ is a certain answer of $q$ in a KB $\mathcal{K}$ if it is an answer in every model of $\mathcal{K}$. If a query does not contain any answer variables, it is a Boolean CQ, and we say it is entailed by a KB $\mathcal{K}$ (interpretation $\mathcal{I}$) if it has the empty vector as answer. The complexity of query entailment for KBs in various DLs is shown in the left-part of Figure[1].

Probabilistic Complexity Classes. The complexity class PP is defined using probabilistic Turing machines, which are like non-deterministic Turing machines, but with an alternative acceptance condition: namely, they accept an input iff at least half of the computation paths end in an accepting state. PP describes the class of all problems that can be decided by a probabilistic Turing machine in which all paths are polynomially bounded by the size of the input. By using oracles, we can obtain the complexity classes PP$^NP$ and P$^{PP}$, for which we have the relations NP $\cup$ coNP $\subseteq$ PP $\subseteq$ PP$^NP$ $\subseteq$ P$^{PP}$ $\subseteq$ PSPACE [33]. Strongly related to the decision class PP is the function class #P, which is the class of functions that can be computed by counting accepting paths in a non-deterministic polynomial time-bounded Turing machine.

3 Temporal Probabilistic Knowledge Bases and Queries

We introduce our framework for temporal probabilistic query answering.

Temporal Probabilistic Knowledge Bases. Regarding the probabilistic aspects, we follow the paradigm introduced in [24] for atemporal probabilistic KBs. To keep things simple, we focus on the easiest type of probabilistic fact bases presented there, the so-called assertion-independent probabilistic ABoxes (ipABoxes). Here, assertions are assigned probabilities, which are assumed to be statistically independent. They correspond to the tuple-independent probabilistic databases studied in [16].

An ipBox is a set of probabilistic ABox assertions of the form $\alpha: p$, where $\alpha$ is an ABox assertion and $p$ is a probability value between 0 and 1. Intuitively, $\alpha: p$ expresses that the assertion $\alpha$ holds with a probability of at least $p$. Instead of $\alpha: 1$, we may just write $\alpha$ if the meaning is clear from the context.

ipABoxes only specify a lower bound on the probability, to conform with the open-world semantics common in ontology-based representations. This means, we might infer using other information in the KB that the probability is in fact higher. A temporal probabilistic KB (TPKB) is now a tuple $(\mathcal{T}, \{\mathcal{A}_i\}_{i \in [1, \lfloor n \rfloor]})$, where $\mathcal{T}$ is a TBox and $\{\mathcal{A}_i\}_{i \in [1, \lfloor n \rfloor]}$ is a sequence of ipABoxes.

We define the semantics of TPKBs using the possible worlds semantics, as common to probabilistic logics and databases [16]. For a given a TPKB $\mathcal{K} = (\mathcal{T}, \{\mathcal{A}_i\}_{i \in [1, \lfloor n \rfloor]})$, the set $\Omega_\mathcal{K}$ of possible worlds of $\mathcal{K}$ contains all sequences $w = (\mathcal{A}_i')_{i \in [1, \lfloor n \rfloor]}$ of classical ABoxes such that for every $i \in [1, \lfloor n \rfloor]$ and $\alpha \in \mathcal{A}_i'$, $\mathcal{A}_i'$ contains an axiom of the form $\alpha: p$. Each TPKB uniquely defines a probability space $(\Omega_\mathcal{K}, \mu_\mathcal{K})$, where the probability measure $\mu_\mathcal{K} : 2^{\Omega_\mathcal{K}} \rightarrow [0, 1]$ satisfies

$$\mu_\mathcal{K}((\{\mathcal{A}_i'\}_{i \in [1, \lfloor n \rfloor]}) = \prod_{i \in [1, \lfloor n \rfloor]} p \cdot \prod_{\alpha \in \mathcal{A}_i'} (1 - p)$$

Footnote: This is different to the open-world semantics for probabilistic databases suggested in [13], which assumes a fixed upper probability for facts absent in the data.
and for $W \subseteq \Omega_{\mathcal{K}}$, $\mu_{\mathcal{K}}(W) = \sum_{w \in W} \mu(\{w\})$. Intuitively, $\mu_{\mathcal{K}}(W)$ gives the probability of being in one of the possible worlds in $W$, by summing up the probabilities of each possible world. The definition of $\mu_{\mathcal{K}}(W)$ reflects the assumption that all probabilities in the TPKB are statistically independent.

**Example 1.** We define the TPKB $\mathcal{K} = \langle \mathcal{T}, (\mathcal{A}_i)_{i \in [1,5]} \rangle$, where $\mathcal{T}$ contains the axiom

$$\text{HighBloodPressurePatient} \equiv \exists \text{hasBloodPressure}. \text{HighBloodPressure},$$

and the probabilistic ABoxes are

$$\mathcal{A}_1 = \{\text{hasBloodPressure}(a, b), \text{HighBloodPressure}(b):0.7\}$$
$$\mathcal{A}_3 = \{\text{HighBloodPressure}(b):0.9\}$$
$$\mathcal{A}_4 = \{\text{HighBloodPressure}(b):0.6\}$$

and $\mathcal{A}_2 = \mathcal{A}_5 = 0$. Every possible world $w = (\mathcal{A}_i)_{i \in [1,5]}$ with $\text{hasBloodPressure}(a, b) \not\in \mathcal{A}_i'$ has probability $\mu_{\mathcal{K}}(w) = 0$. The remaining possible worlds, excluding time points 2 and 5, are shown in Figure 1, with the probability measure $\mu_{\mathcal{K}}$ shown in the last column, where hBP is short for hasBloodPressure and hBP is short for HighBloodPressure.

**Remark 1.** We follow the semantical idea of ipABoxes, in which all assertions are assumed to be statistically independent, mainly to keep the representation simple. Of course, for realistic applications, the assumption that all probabilistic facts are statistically independent is not always accurate, and already [24] specify a more general concept of probabilistic ABoxes. The complexity upper bounds established in this paper only rely on a fixed probability distribution over the possible worlds, which is why they can be easily extended to more refined settings.

<table>
<thead>
<tr>
<th>$\Omega_{\mathcal{K}}$</th>
<th>$\mathcal{A}_1$</th>
<th>$\mathcal{A}_3$</th>
<th>$\mathcal{A}_4$</th>
<th>$\mu_{\mathcal{K}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>hBP(p, b), HBP(b)</td>
<td>HBP(b)</td>
<td>HBP(b)</td>
<td>0.378</td>
</tr>
<tr>
<td>$w_2$</td>
<td>hBP(p, b)</td>
<td>HBP(b)</td>
<td>HBP(b)</td>
<td>0.162</td>
</tr>
<tr>
<td>$w_3$</td>
<td>hBP(p, b), HBP(b)</td>
<td>$\emptyset$</td>
<td>HBP(b)</td>
<td>0.042</td>
</tr>
<tr>
<td>$w_4$</td>
<td>hBP(p, b)</td>
<td>$\emptyset$</td>
<td>HBP(b)</td>
<td>0.018</td>
</tr>
<tr>
<td>$w_5$</td>
<td>hBP(p, b), HBP(b)</td>
<td>HBP(b)</td>
<td>$\emptyset$</td>
<td>0.252</td>
</tr>
<tr>
<td>$w_6$</td>
<td>hBP(p, b)</td>
<td>HBP(b)</td>
<td>$\emptyset$</td>
<td>0.108</td>
</tr>
<tr>
<td>$w_7$</td>
<td>hBP(p, b), HBP(b)</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>0.028</td>
</tr>
<tr>
<td>$w_8$</td>
<td>hBP(p, b)</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>0.012</td>
</tr>
</tbody>
</table>

Table 1: Probability space of example TPKB.

Based on the probability measure, we define models by assigning to each possible world a sequence of classical interpretations. A model of a TPKB $\mathcal{K} = \langle \mathcal{T}, (\mathcal{A}_i)_{i \in [1,n]} \rangle$ is a mapping $\iota$ from possible worlds $w = (\mathcal{A}_i')_{i \in [1,n]} \in \Omega_{\mathcal{K}}$ to sequences $(\iota(w))_{i > 0}$ of (classical) models of $\mathcal{T}$ s.t. for all $i \in [1,n]$, $\iota(w)_i$ is a model of the classical knowledge base $\langle \mathcal{T}, \mathcal{A}_i' \rangle$, and all $\iota(w)_i$ have the same set $\Delta^i$ of domain elements (constant domain assumption).

**Rigid Names.** As typical for temporal knowledge bases, we may assume in addition a set $N_{\text{rig}}$ of rigid names, containing the set $N_{\text{crig}} \subseteq N_{\mathcal{K}}$ of rigid concept names and the set $N_{\text{rig}} \subseteq N_{\mathcal{K}}$ of rigid role names. Rigid names denote names whose interpretation is independent of the flow of time. We say that a model $\iota$ of a TPKB $\mathcal{K} = \langle \mathcal{T}, (\mathcal{A}_i)_{i \in [1,n]} \rangle$ respects rigid names iff for all $w \in \Omega_{\mathcal{K}}$, $i, j \in [1,n]$ and $X \in N_{\text{rig}}$, $X(i(w))_i = X(i(w))_j$. Allowing for rigid names often has a direct impact on complexity and decidability of common reasoning problems, which is why typically different cases based on whether $N_{\text{crig}} = \emptyset$ or $N_{\text{rig}} = \emptyset$ are studied for complexity.
we might be interested only in answers that hold with a certain minimal or maximal probability. Rather

The propositional operators, to be able to describe all this, we extend TQs with probability operators. A

Temporal Probabilistic Queries. To query temporal data in the OBDA framework, extensions of

temporal probabilistic KBs, while the concept HighBP is not rigid, as the blood pressure of a patient can change from high
to not high. As a consequence, the individual p will be related to the blood pressure b at all time points,
even though the assertion hasBP(p, b) only occurs in A1.

Example 2. In the above example, the relation hasBP is rigid, as its interpretation should be independent
time, while the concept HighBP is not rigid, as the blood pressure of a patient can change from high
to not high. As a consequence, the individual p will be related to the blood pressure b at all time points,
even though the assertion hasBP(p, b) only occurs in A1.

Example 2. In the above example, the relation hasBP is rigid, as its interpretation should be independent
of time, while the concept HighBP is not rigid, as the blood pressure of a patient can change from high
to not high. As a consequence, the individual p will be related to the blood pressure b at all time points,
even though the assertion hasBP(p, b) only occurs in A1.

To be able to describe all this, we extend TQs with probability operators. A temporal probabilistic
query (TPQ) is one of the following forms, where q is a CQ, φ1 and φ2 are TPQs, p ∈ [0, 1] and

The propositional operators, (next), (eventually), U (until) are found in TQs, while P∼p is the
operator that we add to this language. Note that due the disjunction operator, we can also express
unions of conjunctive queries (UCQs), which are simply disjunctions of CQs. The answer variables of a TPQ
φ are the answer variables of the CQs in φ. A TPQ φ is Boolean if every variable in φ is bound by an
existential quantifier.

In order to define the semantics of TPQs, we have to take into consideration the two dimensions in which

| φ            | t, w, i |= φ iff |
|--------------|----------------|
| ∼y, ψ(y)     | t(w), i |= ∼y, ψ(y) |
| ¬φ1          | t, w, i ¬|= φ1 |
| φ1 ∧ φ2      | t, w, i |= φ1 and t, w, i |= φ2 |
| φ1 ∨ φ2      | t, w, i |= φ1 or t, w, i |= φ2 |
| ○φ1          | t, w, i + 1 |= φ1 |
| ○¬φ1         | t, w, i − 1 |= φ1 and i > 0 |
| ◊φ1          | t, w, j |= φ1 for some j ≥ i |
| ◊¬φ1         | t, w, j |= φ1 for some j ≤ i |
| □φ1          | t, w, j |= φ1 for all j ≥ i |
| □¬φ1         | t, w, j |= φ1 for all j ≤ i |
| φ1 U φ2      | t, w, j |= φ2 for some j ≥ i, and t, w, k |= φ1 for all k ∈ [i, j − 1] |
| φ1 S φ2      | t, w, j |= φ2 for some j ≤ i, and t, w, k |= φ1 for all k ∈ [j + 1, i] |
| P∼pφ         | µK(κ⇒(w' ∈ ΩK | t, w', i |= φ)) ~ p, where ∼ ∈ {<, ≤, =, ≥, >} |
queries refer to a temporal probabilistic model. While the temporal operators refer to the time line in a single possible world, for the probability operators we have to aggregate the probabilities of the possible worlds in which a query is entailed.

Let $\mathcal{K}$ be a TPKB, $\iota$ a model of $\mathcal{K}$, and $\phi$ a Boolean TPQ. For a single possible world $w \in \Omega_\mathcal{K}$ and a time point $i$, we say that $\phi$ is satisfied at $w, i$ under $\iota$, in symbols $\iota, w, i \models \phi$ iff the conditions in Table 2 are satisfied. Note that the temporal operators refer to the time line of a single possible world, for which they are defined as in [8]. In contrast, the probability operators refer to the current time point in multiple possible worlds, and are defined similar to the probabilistic concept constructor in Prob-$\text{ALC}$ [2]. A Boolean TPQ $\phi$ is satisfied in an interpretation $\iota$ at $i$, in symbols $\iota, i \models \phi$, iff $\iota, w, i \models \phi$ for all $w \in \Omega_\mathcal{K}$. It is entailed by the TPKB $\mathcal{K}$ at $i$, in symbols $\mathcal{K}, i \models \phi$, iff $\iota, w, i \models \phi$ for all models $\iota$ of $\mathcal{K}$. $\phi$ is satisfiable in $\mathcal{K}$ at $i$ iff there exists a model $\iota$ of $\mathcal{K}$ s.t. $\iota, i \models \phi$. Note that satisfiability is complementary to entailment: namely, $\phi$ is satisfiable in $\mathcal{K}$ at $i$ iff $\mathcal{K}, i \not\models \neg \phi$.

Now given a TPKB $\mathcal{K}$, a TPQ $\phi$ with answer variables $\vec{x}$, a time point $i > 0$, and a mapping $\sigma : \vec{x} \rightarrow \mathbb{N}_i$, $\sigma$ is a certain answer for $\phi$ in $\mathcal{K}$ at $i$ iff $\mathcal{K}, i \models \phi'$, where $\phi'$ is the result of applying $\sigma$ on $\phi$. As common, since computing answers for TPQs can be seen as a search problem that uses Boolean TPQ entailment, we focus on the decision problem of query entailment, and may refer to Boolean TPQs simply as TPQs.

Example 3. We consider a slight variation of the query from the introduction.

$$P_{>0.5}(\Diamond \neg \Box (\text{HighBPP}(x) \land \Diamond \text{HighBPP}(x)))$$

For $x = p$ and time point 5, the query below the probability operator is entailed in every model of the possible worlds $w_1, w_2, w_3$ and $w_5$, which together have a probability of 0.834. Consequently, $p$ is an answer to the query at time point 5. Now consider the variation where the probability operators are moved inside.

$$\Diamond \neg \Box (P_{>0.5}(\text{HighBPP}(x))) \land \Diamond P_{>0.5}(\text{HighBPP}(x)))$$

This corresponds to the situation where at least twice in the last 5 time units, the probability of having a high blood pressure was above 0.8. As this probability is only once above this bound, this query is not entailed.

The complexity of TPQ entailment for various DLs is shown in Figure 1 where we compare against the complexity of classical query entailment (left column), and distinguish the cases based on whether $N_{\text{rig}} = \emptyset$ (middle column) or $N_{\text{rig}} \neq \emptyset$ (right column). All complexities remain tight independent on whether we admit rigid concept names ($N_{\text{rig}} \neq \emptyset$). The EXPSPACE-result for $\mathcal{ELH}$ remains tight even without TBoxes, so that any logic below $\mathcal{ELH}$ is covered. Results marked with (pos) regard positive TPQs, which we discuss towards the end of the paper.

4 Hardness of TPQ Entailment

We show that TPQ satisfiability, and thus entailment, is EXPSPACE-hard even if $T = \emptyset$ and $N_{\text{rig}} = N_{\text{rig}} \neq \emptyset$, by reduction of the exponential corridor tiling problem [44]. In this problem, we are given a set $T$ of tile types, two special tile types $t_s, t_e \in T$, a natural number $n$, and two functions $v$ and $h$ of compatibility constraints $v : T \rightarrow 2^T$ (vertical) and $h : T \rightarrow 2^T$ (horizontal). The input is an instance of the exponential corridor tiling problem if there exists a number $m \in \mathbb{N}$ and a tiling $f : [0, m] \times [0, 2^n - 1] \rightarrow T$ such that $f(0, 0) = t_s, f(m, 0) = t_e$, and for all $x \in [0, m]$ and $y \in [0, 2^n - 1]$, if $x < m, f(x + 1, y) \in h(f(x, y))$ and if $y < 2^n - 1, f(x, y + 1) \in v(f(x, y))$.

We use $n$ concept names $A_1$ to mark the different possible worlds $w \in \Omega_\mathcal{K}$ with a counter, such that in interpretations $\iota$ that satisfy both the TPQ and the TPKB, $\iota, w, j \models A_1(a)$ iff the $i$th bit of the counter is 1 at time point $j$, and $\iota, w, j \not\models A_1(a)$ iff the $i$th bit is 0 at time point $j$. The ipABox $A_1 = \{A_1(a) \sim 0.5 \mid i \in [1, n]\}$ assigns every possible world a different counter value. Our query
We show the complexity upper bounds for general queries shown in Figure 1, where we again focus on which we call quasi-models. As we can only check the vertical tiling conditions for one pair of rows at a time, we represent each

At each time point, two possible worlds can be recognised by simple queries: the one whose counter value is 0 (which satisfies $\bigwedge_{1 \leq i \leq n} \neg A_i(a)$), and the one whose counter value is $2^n - 1$ (which satisfies $\bigwedge_{1 \leq i \leq n} A_i(a)$). Unless the latter one represents the last row, these worlds correspond to neighbours in the tiling, which means that for these worlds, we can enforce the vertical tiling conditions with the following query, where $L(a)$ is an assertion that marks the last row, and for a tile type $t \in T$, $B_t(a)$ expresses that the current cell has a tile of type $t$.

$$\Box \bigwedge_{t_1 \in T} \left( B_{t_1}(a) \land \bigwedge_{i \in [1,n]} A_i(a) \land \neg L(a) \right) \rightarrow \bigvee_{t_2 \in \nu(t_1)} P_{=1} \left( \left( \bigwedge_{i \in [1,n]} \neg A_i(a) \right) \rightarrow B_{t_2}(a) \right)$$

As we can only check the vertical tiling conditions for one pair of rows at a time, we represent each cell by up to $2^n$ succeeding time points in each possible world, switching to the next tile only when the counter reaches $2^n - 1$. The remaining details of the reduction can be found in the appendix. The hardness for $\mathcal{ALC}$ with rigid roles follows from the non-probabilistic case [3].

**Theorem 4.** The lower bounds regarding general TPQs in Figure 1 hold.

## 5 Deciding TPQ Entailment

We show the complexity upper bounds for general queries shown in Figure 1, where we again focus on the complementary problem of query satisfiability.

The main idea is to define appropriate abstractions of models of the TPKB $\mathcal{K} = \langle T, (A_i)_{i \in [1,n]} \rangle$ which we call quasi-models, and then show how an abstraction that witnesses the satisfiability can be guessed and verified within the targeted complexity bound. We first define the structure to represent single time points, which we call quasi-states. We can assume without loss of generality that $\phi$ contains only the operators $\land$, $\neg$, $\cup$, $S$ and $P_{=p}$, since the remaining operators can be linearly encoded using known equivalences. Denote by $\text{sub}(\phi)$ the sub-queries of $\phi$ and set $T(\phi) = \{ \psi, \neg \psi \mid \psi \in \text{sub}(\phi) \}$. A quasi-state is now a mapping $Q : \Omega_{\mathcal{K}} \rightarrow T(\phi)$ that satisfies the following conditions:

- **S1** $\neg \psi \in Q(w)$ iff $\psi \notin Q(w)$,
- **S2** for all $\psi_1 \land \psi_2 \in T(\phi)$: $\psi_1 \land \psi_2 \in Q(w)$ iff $\psi_1 \in Q(w)$ and $\psi_2 \in Q(w)$, and
- **S3** for all $P_{=p}(\psi) \in T(\phi)$: $P_{=p}(\psi) \in Q(w)$ iff $\mu_{\mathcal{K}}(\{ w \mid \psi \in Q(w) \}) \sim p$.
The quasi-state abstracts probabilistic interpretations at a single time point by assigning queries to each possible world according to the semantics of the atemporal operators in our query language. To incorporate the temporal dimension, we consider unbounded sequences of quasi-states \((Q_i)_{i \geq 1}\), which we call quasi-models for \(\mathcal{K}\), and which have to satisfy the following conditions for \(i \geq 1\) and \(w = (A_i')_{i \in [1,n]} \in \Omega_{\mathcal{K}}\):

**Q1** if \(i \in [1,n]\), then \(\langle T, A'_i \rangle \not\models \neg \left(\bigwedge_{\psi \in X} \psi \right)\), where \(X = \{\psi \in Q_i(w) \mid \psi \text{ is a CQ or a negated CQ}\}\).

**Q2** for all \(\bigcirc \psi \in T(\phi), \bigcirc \psi \in Q_i(w)\) iff \(\psi \in Q_{i+1}(w)\).

**Q3** for all \(\bigcirc \neg \psi \in T(\phi), \bigcirc \neg \psi \in Q_i(w)\) iff \(\psi \in Q_{i+1}(w)\).

**Q4** for all \(\psi_1 U \psi_2 \in T(\phi), \psi_1 U \psi_2 \in Q_i\) iff there exists \(j \geq i\) s.t. \(\psi_2 \in Q_j(w)\) and for all \(k \in \llbracket j-1, i \rrbracket\), \(\psi_1 \in Q_k(w)\), and

**Q5** for all \(\psi_1 S \psi_2 \in T(\phi), \psi_1 S \psi_2 \in Q_i\) iff there exists \(j \leq i\) s.t. \(\psi_2 \in Q_j(w)\) and for all \(k \in \llbracket j-1, i \rrbracket\), \(\psi_1 \in Q_k(w)\).

Again, the intuition behind these conditions is given directly by the semantics of the temporal operators.

To handle rigid names, we need an additional structure to make sure that the queries assigned to different time points in a possible world correspond to a sequence of interpretations that respects rigid names. Let \(\{q_1, \ldots, q_m\}\) be the CQs that occur in the query \(\phi\). For each \(w \in \Omega_{\mathcal{K}}\), we guess the set \(S(w) \subseteq 2^{\{q_1, \ldots, q_m\}}\) of sets of queries that are allowed be satisfied together at a time point in \(w\), and thus obtain a mapping \(S : \Omega_{\mathcal{K}} \rightarrow 2^{\{q_1, \ldots, q_m\}}\). Intuitively, for each time point \(i \geq 0\), the set of CQs included in \(Q_i(w)\) has to be some element of \(S(w)\). To also take into account the ABoxes, we use a second mapping \(a : \Omega_{\mathcal{K}} \times \llbracket 1, n \rrbracket \rightarrow 2^{\{q_1, \ldots, q_m\}}\), which for each \(w \in \Omega_{\mathcal{K}}\) assigns elements from \(S(w)\) to the ABoxes in \(w\). Given such mappings \(S\) and \(a\), we say that a quasi-model \((Q_i)_{i \geq 1}\) is compatible to \(S\) and \(a\) if for every \(i \geq 0\) and \(w \in \Omega_{\mathcal{K}}\):

**Q6** \(Q_i(w) \cap \{q_1, \ldots, q_m\} \subseteq S(w)\), and

**Q7** if \(i \in [1, n]\), \(Q_i(w) \cap \{q_1, \ldots, q_m\} = a(w,i)\).

The following definition captures when \(S\) and \(a\) correspond to a model of \(\mathcal{K}\) that respects rigid names.

**Definition 5.** Let \(w = (A'_i)_{i \in [1,n]} \in \Omega_{\mathcal{K}}\), \(S : \Omega_{\mathcal{K}} \rightarrow 2^{\{q_1, \ldots, q_m\}}\) and \(a : \Omega_{\mathcal{K}} \times \llbracket 1, n \rrbracket \rightarrow 2^{\{q_1, \ldots, q_m\}}\), where \(S(w) = \{X_1, \ldots, X_k\}\). Then, \(S\) is called r-satisfiable wrt. \(w\) and \(a\) iff there exist (classical) interpretations \(I_1, \ldots, I_k, I_1, \ldots, I_n\) such that

**R1** the interpretations are models of \(T\),

**R2** for any two interpretations \(I, I' \in \{I_1, \ldots, I_k, I_1, \ldots, I_n\}\), we have \(\Delta^I = \Delta^{I'}\) and \(X^I = X^{I'}\) for all \(X \in N_{rig}\),

**R3** for all \(i \in [1, k]\), \(I_i \models \bigwedge_{q \in X_i} q \land \bigwedge_{q \notin X_i} \neg q\), and

**R4** for all \(i \in [1, n]\), \(I_i \models \bigwedge_{q \in a(w,i)} q \land \bigwedge_{q \notin a(w,i)} \neg q\) and \(I_i \models A'_i\).

\(S\) is r-satisfiable wrt. \(a\), if for all \(w \in \Omega_{\mathcal{K}}\), \(S\) is r-satisfiable wrt. \(w\) and \(a\).

Note that the interpretations \(I_1, \ldots, I_k\) in the interpretation correspond to the elements \(\{X_1, \ldots, X_k\} = S(w)\), so that Condition \(\text{R2}\) ensures that we can find sequences of interpretation that respect rigid names.
Lemma 6. Wrt. the size of $\mathcal{K}$ and $\phi$, r-satisfiability for $\mathcal{L}$-TPKBs can be decided in

1. NExpTime for $\mathcal{L} = \mathcal{ELH}$,
2. NExpTime for $\mathcal{L} = \mathcal{SHQ}$ if $N_{\text{rig}} = \emptyset$,
3. 2-ExpTime for $\mathcal{L} \in \{ \mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHOI} \}$, and
4. it is decidable for $\mathcal{L} = \mathcal{ALCHOIQ}$.

Proof (Sketch). We define a classical KB based on the mappings $a$ and a world $w \in \Omega_{\mathcal{K}}$ which encodes the interpretation of non-rigid names $Y \in (N_C \cup N_R) \setminus N_{\text{rig}}$ for different elements $X_i \in S_i(w)$ using fresh names $Y_i$. A similar translation is applied to the CQs $q \in X_i$. We can then reduce the properties in Definition 5 to a query entailment problem, where the KB and the query are of exponential size with respect to the input. While query entailment for $\mathcal{ALCI}$ and $\mathcal{ALCO}$ is 2-ExpTIME-hard [27, 30], we obtain by inspection of the procedures in [22, 21, 11] that this particular query entailment test can be performed in 2-ExpTIME. For $\mathcal{SHQ}$ and $N_{\text{rig}} = \emptyset$, the complexity follows from results in [3].

Quasi-models are indeed sufficient to witness the satisfiability of a TPQ. If the quasi-model is additionally compatible to mappings $S$ and $a$ s.t. $S$ is r-satisfiable wrt. $a$, then they witness the satisfiability of a TPQ from TKBs with rigid names. Crucially for our complexity result, it is further sufficient to focus on quasi-models that have a regular shape.

Lemma 7. $\phi$ is satisfiable in $\mathcal{K}$ at time point $i$ iff there exists mappings $S : \Omega_{\mathcal{K}} \rightarrow 2^{(q_1, \ldots, q_m)}$ and $a : \Omega_{\mathcal{K}} \times [1, n] \rightarrow 2^{(q_1, \ldots, q_m)}$ and a quasi-model $(Q_j)_{j \geq 1}$ for $\mathcal{K}$ such that

1. $\phi \in Q_i(w)$ for all $w \in \Omega_{\mathcal{K}}$,
2. $(Q_j)_{j \geq 1}$ is compatible with $S$ and $a$,
3. $S$ is r-satisfiable wrt. to $a$, and
4. $(Q_j)_{j \geq 1}$ is of the form $Q_1, \ldots, Q_m(Q_{m+1}, \ldots Q_{m+o})^\omega$, where $m$ and $o$ are both double exponentially bounded in the size of $\mathcal{K}$.

Theorem 8. The complexity upper bounds for general TPQs in Figure 7 hold.

Proof (Sketch). We first guess the numbers $m$ and $o$ from Lemma 7. If $N_{\text{rig}} \neq \emptyset$, we additionally guess the mappings $S$ and $a$ and verify that $S$ is r-satisfiable wrt. $a$. We now guess the quasi-states $Q_1, \ldots, Q_{m+o}$ one after the other, where we carefully make sure that all the conditions in the definition of quasi-states and quasi-models are satisfied, and verify that $Q_{m+o}$ is compatible to $Q_{m+1}$. This procedure runs in exponential space if r-satisfiability can be decided in exponential space, and in double exponential time if deciding r-satisfiability requires double exponential time.

6 Positive TPQs

It turns out that for $\mathcal{EL}$, we can obtain better complexity bounds if we restrict ourselves to positive TPQs, which are TPQs that do not use the operators $\neg$, $P_{<p}$ and $P_{=p}$. The probability operators $P_{<p}$ and $P_{=p}$ can be seen as implicit negation operators, as they express the non-entailment of a query $\phi$ in some possible worlds, whereas $P_{<p}\phi$ only expresses the positive entailment of $\phi$ in some possible worlds. The examples used in this paper all use only positive TPQs.

Definition 9. A TPQ is positive iff it does not use the operators $\neg$, $P_{<p}$, $P_{\leq p}$ and $P_{=p}$.
For DLs that have negation, our reduction used to show \textsc{ExpSpace}-hardness can be adapted to query entailment for positive TPQs. As we reduced the corridor tiling problem to query satisfiability, the corresponding query entailment problem is of the form $K \models \neg \phi$, where $\phi$ is the defined query. By pushing negations inside, we obtain a query in which every probability operator is of the form $P_{>0}$ or $P_{\geq 1}$, and negation only occurs in front of concept names. Therefore, for any DL extending $\mathcal{ALC}$, the complexity bounds established in the last sections remain tight even for positive TPQs. In contrast to $\mathcal{ALC}$, $\mathcal{EL}$ has the canonical model property, which makes it possible to test for entailment in different possible worlds independently. This allows for a strategy in which the TPQ is evaluated “inside out”, by first evaluating the most nested probability operators, and then proceeding on the next level. Due to the known closure properties of the complexity class $\mathcal{PP}$, we obtain a $\mathcal{PP} \cap \mathcal{NP}$ complexity upper bound if the nesting depth of the probability operators is bound, which we show to be tight, and otherwise a $\mathcal{P} \cap \mathcal{NP}$ upper bound. This approach further allows us to establish tight complexity for data complexity, where the size of the query is assumed to be fix, marked in Figure 1 with (pos,dat).

**Theorem 10.** The complexity results regarding positive TPQs in Figure 1 hold.

### 7 Conclusion

We introduced a framework for representing and querying temporal probabilistic data within the ontology-based query answering paradigm, and established tight complexity bounds for most common description logics. While for expressive DLs starting from $\mathcal{ALCIT}$ and $\mathcal{ALCO}$, adding both the temporal and the probabilistic dimension comes at no additional cost compared to classical query answering, for $\mathcal{ALC}$ and below, reasoning becomes harder both in comparison to purely temporal and purely probabilistic query answering. For instance, probabilistic query answering is $\mathcal{ExpTime}$-complete for $\mathcal{ALC}$ and $\mathcal{PP} \cap \mathcal{NP}$-complete for $\mathcal{EL}$, and for $\mathcal{N_{Rig}} = \emptyset$, it is $\mathcal{ExpTime}$-complete for $\mathcal{ALC}$ and $\mathcal{PSPACE}$-complete for $\mathcal{EL}$, which contrasts with our \textsc{ExpSpace}-hardness that occurs already without a TBox. For $\mathcal{EL}$, this situation can be improved if we forbid negation in the query language, in which case temporal probabilistic query answering is not harder as in the atemporal case. We believe that our technique for showing the upper bound here could also be used in practical implementations. An open question this approach could be used in conjunction with query rewriting techniques for simpler DLs such as $\mathcal{DL-Lite}$, on probabilistic database systems.

### 8 Acknowledgements

This work is supported by the German Research Foundation (DFG) within the Collaborative Research Center SFB 912 HAEC.

### References


A Proofs of Theorems and Lemmata

We provide proofs for the theorems and lemmata in the paper. Note that this document has an extended bibliography section at the end.
A.1 Lower Bounds

Theorem 4. The lower bounds regarding general TPQs in Figure 1 hold.

Proof. For DLs extending $\text{ALCI}$ or $\text{ALCO}$, $2\text{-ExpTime}$ lower bounds follow from the corresponding results in classical query entailment [27, 30].

For the remaining DLs, we provide a reduction of the $\text{ExpSpace}$-complete $2^n$ corridor tiling problem as specified in the main text. We provide an encoding of this problem using a single ipABox $\mathcal{A}$ and a TPCQ $\phi$ with negations. $\mathcal{A}$ and $\phi$ are constructed in such a way that there is a correspondence between solutions $f$ to the tiling problem and models $\iota$ of $\mathcal{K} = (\emptyset, (\mathcal{A}))$ and $\phi$. The (bounded) vertical dimension of the corridor is represented across the $2^n$ possible worlds, while the (unbounded) horizontal dimension is represented along the time line. Specifically, the correspondence from $\iota$ to a tiling $f$ is specified via two mappings:

1. $y : [0, 2^n - 1] \rightarrow \Omega_\mathcal{K}$ maps each $y$-coordinate of the tiling to a possible world
2. $c : \mathbb{N} \times \Omega_\mathcal{K} \rightarrow T$ maps each pair of $x$-coordinate and possible world to the tile type.

The query will then be constructed in such a way that, in case of satisfiability, we can obtain the tiling $f(i, j) = c(2^n \cdot j + i, y(j))$.

We use a concept name $B_t$ for every tile type $t \in T$, and use the assertion $B_t(a)$ to denote that the cell corresponding to a possible world/time point pair has a tile of type $t$. To make sure that every pair of a possible world and a time point represents exactly one $t \in T$, we use the query

\[
\Box \bigwedge_{i \in T} \left( B_{t_1}(a) \leftrightarrow \bigwedge_{t_2 \in T, t_1 \neq t_2} \neg B_{t_2}(a) \right).
\]

We can already define the first mapping $c$ from pairs of possible worlds and time points to their tile type: given a model $\iota$ of the final query, we define $c(i, w) = t$, where $\iota, w, i \models B_t(a)$.

To provide for the mapping $y : [0, 2^n - 1] \rightarrow \Omega_\mathcal{K}$, which assigns row numbers to possible worlds, we set

\[
\mathcal{A}_1 = \{ A_i(a) : 0.5 \mid i \in [1, n] \},
\]

where $A_1, \ldots, A_n$ are concept names that correspond to bits in a binary counter. Since all probabilities are statistically independent, each counter value is represented by some possible world. For $i \in [0, 2^n - 1]$, the value of $y(i)$ is then simply the possible world in which the counter has the value $i$ at the first time point. We mark the possible world which represents the last row with the assertion $L(a)$ using the following queries.

\[
\left( \bigwedge_{i \in [1, n]} A_i(a) \right) \leftrightarrow L(a)
\]

\[
\Box (L(a) \leftrightarrow \Diamond L(a))
\]

The previous queries ensure that the mappings $y$ and $c$ are well-defined. The following query ensures the tiling conditions regarding the special tiles $t_s$ and $t_e$, which have to occur in the first row of respectively the first and the last column of the tiling solution.

\[
\left( \bigwedge_{i \in [1, n]} \neg A_i(a) \right) \rightarrow (B_{t_s}(a) \land B_{t_e}(a))
\]
It remains to provide queries that enforce the tiling conditions. The following queries ensure that the counters in each possible world get incremented across the time line.

\[
\bigwedge_{i \in [1,n]} \left( \neg A_i(a) \land \bigwedge_{j < i} A_j(a) \right) \rightarrow \bigcirc \left( A_i(a) \land \bigwedge_{j < i} \neg A_j(a) \right)
\]

\[
\bigwedge_{i \in [1,n]} \left( \neg A_i(a) \land \bigvee_{j < i} \neg A_j(a) \right) \rightarrow \bigcirc \neg A_i(a)
\]

\[
\bigwedge_{i \in [1,n]} \left( A_i(a) \land \bigvee_{j < i} \neg A_j(a) \right) \rightarrow \bigcirc A_i(a)
\]

\[
\bigcirc \left( \bigwedge_{i \in [1,n]} A_i(a) \right) \rightarrow \bigwedge_{i \in [1,n]} \neg A_i(a)
\]

In each possible world, the current tile type is transported to the next time point until the counter reaches \(2^n - 1\).

\[
\bigwedge_{t \in T} \left( B_t(a) \land \bigvee_{i \in [1,n]} \neg A_i(a) \right) \rightarrow \bigcirc B_t(a)
\]

Note that we only have to check the tiling conditions until the special tile \(t_e\) has been placed in the first row: afterwards, we do not care. We use the assertion \(B_{end}(a)\) to mark the last column of the tiling:

\[
\bigwedge_{i \in [1,n]} \neg A_i(a) \rightarrow (P_{=1}(\neg B_{end}(a)) \cup (B_{end}(a) \land t_e(a)))
\]

This query ensures that no possible world satisfies \(B_{end}(a)\) until the last tile has been placed in the first row.

As explained in the main text, if the counter in a possible world reaches \(2^n - 1\), we can identify the world that corresponds to the next row easily, as its counter then has the value 0. Note that we only have to enforce the tiling conditions until the last tile has been placed, which is the case as soon as in some possible world the assertion \(B_{end}(a)\) is satisfied. We can thus enforce the vertical tiling conditions using the following query.

\[
\bigwedge_{t_1 \in T} \left( B_{t_1}(a) \land \neg L(a) \land \bigwedge_{i \in [1,n]} A_i(a) \right) \rightarrow \bigvee_{t_2 \in \text{ev}(t_1)} P_{=1} \left( \bigwedge_{i \in [1,n]} \neg A_i(a) \rightarrow B_{t_2}(a) \right) \cup P_{>0}(B_{end}(a))
\]

To enforce the horizontal constraints, we only have to identify the next time point when the counter is \(2^n - 1\).

\[
\bigwedge_{t_1 \in T} \left( B_{t_1}(a) \land \bigwedge_{i \in [1,n]} A_i(a) \right) \rightarrow \bigcirc \bigvee_{t_2 \in \text{h}(t_1)} B_{t_2}(a) \cup P_{>0}(B_{end}(a))
\]
We show both directions of the lemma separately.

We show that \( S \) is r-satisfiable wrt. \( a \), and consequently, by induction, \( S_1 \) operators defined in Table 2 to Conditions \( \mathbf{RT} \) and \( \mathbf{K4} \).

\[ \exists w \in \Omega_K, \ S \text{ is r-satisfiable wrt. } a \text{ and } w, \text{ and} \]

\[ \therefore \text{there exists a quasi-model } (Q_j)_{j \geq 1} \text{ for } \phi \text{ in } K \text{ compatible with } S \text{ and } a \text{ s.t. } \phi \in Q_i(w) \text{ for all } w \in \Omega_K. \]

Lemma 12. If \( \phi \) is satisfiable in \( K \) at \( i \), then there exist mappings \( S : \Omega_K \rightarrow 2^{q_1 \ldots q_m} \) and \( a : \Omega_K \times \{1, n\} \rightarrow 2^{q_1 \ldots q_m} \) s.t.

1. \( S \) is r-satisfiable wrt. \( a \) and \( w \), and

2. there exists a quasi-model \( (Q_j)_{j \geq 1} \) for \( \phi \) in \( K \) compatible with \( S \) and \( a \) s.t. \( \phi \in Q_i(w) \) for all \( w \in \Omega_K \).

Proof. Assume \( \phi \) is satisfiable in \( K \) at \( i \). There then exists a temporal probabilistic model \( \iota \) of \( K \) s.t. for all \( w \in \Omega_K \), \( \iota, w, i \models \phi \). For \( w \in \Omega_K \) and \( j \geq 1 \), set \( X_j(w) = \{ q \text{ is a CQ in } \phi \mid \iota(w)_i = q_i \} \). \( S \) and \( a \) are now defined by setting for all \( w \in \Omega_K \) and \( i \in \{1, n\} \):

\[ S(w) = \{ X_j(w) \mid j \geq 1 \} \]
\[ a(w, i) = X_i(w). \]

It follows by construction that for each \( w \in S, S \) is r-satisfiable wrt. \( a \) and \( w \). Indeed, the interpretations \( I_1, \ldots, I_n, J_1, \ldots, J_n \) that witness this can be directly taken from \( i \) for \( i \in \{1, n\} \), we have \( I_i = \iota(w)_i \), and for each \( X_i \in S(w) \), there exists some \( j \geq 1 \) s.t. \( X_j(w) = X_i \), for which we set \( J_i = \iota(w)_j \).

Conditions \( \mathbf{RT} \) and \( \mathbf{K4} \) are now readily checked.

The quasi-model \( (Q_j)_{j \geq 1} \) is now defined by setting

\[ Q_j(w) = \{ \psi \in T(\phi) \mid \iota, w, j \models \psi \} \]

for all \( w \in \Omega_K \) and \( i > 1 \).

We show that \( (Q_j)_{j \geq 1} \) is indeed a quasi-model compatible with \( S \) and \( a \).

For each \( j \geq 1 \), it follows by direct correspondence between the semantics of the atemporal query operators defined in Table 2 to Conditions \( \mathbf{S1} \) and \( \mathbf{S3} \) that \( Q_j \) is a quasi-state. Specifically, for Condition \( \mathbf{S3} \) we have \( \iota, w, j \models P_{\psi}(\psi) \) iff

\[ \mu_K(\{ w' \in \Omega_K \mid \iota, w', j \models \psi \}) \sim p, \]

and consequently, by induction, \( P_{\psi}(\psi) \in Q_j(w) \) iff

\[ \mu_K(\{ w' \in \Omega_K \mid \psi \in Q_j(w) \}) \sim p. \]
Similarly, \((Q_j)_{j \geq 1}\) satisfies Conditions\(\mathbf{Q2}, \mathbf{Q5}\) by direct correspondence to the semantics of the temporal query operators defined in Table 3. For Condition\(\mathbf{Q1}\), we have to show that for \(w = (\mathcal{A}_i')_{i \in [i]} \in \Omega_K\) and \(i \in [1, n]\),

\[
\langle T, \mathcal{A}_i' \rangle \not\models \neg \left( \bigwedge_{\psi \in X} \psi \right),
\]

where

\[
X = \{ \psi \in Q_i(w) \mid \psi \text{ is a CQ or a negated CQ}\}.
\]

Let \(X = \{ \psi \in Q_i(w) \mid \psi \text{ is a CQ or a negated CQ}\}\). By construction, \(\iota, w, i \models \bigwedge_{\psi \in X} \psi\). Furthermore, since \(i\) is a model of \(K\), \(i(w)_j\) is a model of \(\langle T, \mathcal{A}_i' \rangle\). As a consequence, we cannot have \(\langle T, \mathcal{A}_i' \rangle \models \neg \left( \bigwedge_{\psi \in X} \psi \right)\), and therefore \(\langle T, \mathcal{A}_i' \rangle \not\models \neg \left( \bigwedge_{\psi \in X} \psi \right)\) and \((Q_j)_{j \geq 1}\) satisfies Condition\(\mathbf{Q1}\). We obtain that \((Q_j)_{j \geq 1}\) satisfies Conditions\(\mathbf{Q1}, \mathbf{Q5}\) and thus that \((Q_j)_{j \geq 1}\) is a quasi-model.

It remains to show that \((Q_j)_{j \geq 1}\) is compatible with \(S\) and \(a\), that is, it satisfies Conditions\(\mathbf{Q6}, \mathbf{Q7}\). Let \(j \geq 1\) and \(w \in \Omega_K\).

\(\mathbf{Q6}\) We have to show that \(Q_j(w) \cap \{q_1, \ldots, q_m\} \in \mathcal{S}(w)\). Let \(X = Q_j(w) \cap \{q_1, \ldots, q_m\}\). By construction of \(Q_j(w)\), \(X\) contains exactly the CQs from \(\{q_1, \ldots, q_m\}\) that are entailed by \(\iota(w)_j\). Consequently, \(X = X_j(w)\). We have \(X_j(w) \in \mathcal{S}(w)\) by construction of \(\mathcal{S}(w)\), and therefore, \(X \in \mathcal{S}(w)\).

\(\mathbf{Q7}\) We have to show that if \(j \in [1, n]\), then \(Q_j(w) \cap \{q_1, \ldots, q_m\} = a(w, j)\). Let \(j \in [1, n]\) and \(X = Q_j(w) \cap \{q_1, \ldots, q_m\}\). By construction of \(Q_j(w)\), \(X\) contains exactly the CQs from \(\{q_1, \ldots, q_m\}\) that are entailed by \(\iota(w)_j\), and therefore, \(X = X_j(w)\). By definition of \(a\), we have \(a(w, j) = X_j(w)\), and therefore \(a(w, j) = X\).

We established that \(S\) is \(r\)-satisfiable wrt. \(a\), and that \((Q_j)_{j \geq 1}\) is a quasi-model compatible to \(S\) and \(a\). Furthermore, we have \(\phi \in Q_i(w)\) for all \(w \in \Omega_K\), since by assumption, \(\iota, w, i \models \phi\) for all \(w \in \Omega_K\). It follows that Conditions 1 and 2 from the Lemma are satisfied.

**Lemma 13.** Let \(\mathcal{S} : \Omega_K \to 2^{\{q_1, \ldots, q_m\}}\) and \(a : \Omega_K \times [1, n] \to 2^{\{q_1, \ldots, q_m\}}\) be such that

1. for every \(w \in \Omega_K\), \(\mathcal{S}\) is \(r\)-satisfiable wrt. \(a\) and \(w\), and
2. there exists a quasi-model \((Q_j)_{j \geq 1}\) for \(\phi\) in \(K\) compatible with \(\mathcal{S}\) and \(a\) s.t. \(\phi \in Q_i(w)\) for all \(w \in \Omega_K\).

Then, \(\phi\) is satisfiable in \(K\) at \(i\).

**Proof.** Assume there exist the mappings \(\mathcal{S}\) and \(a\), as well as a quasi-model \((Q_j)_{j \geq 1}\), as in the lemma. We construct a temporal probabilistic model \(\iota\) of \(K\) s.t. for all \(w \in \Omega_K\), \(\iota, w, 1 \models \phi\). For all \(w \in \Omega_K\), assume \(\mathcal{S}(w) = \{X^w_1, \ldots, X^w_{m_w}\}\). By Definition 5 of \(r\)-satisfiability, for every \(w \in \Omega_K\), there exists interpretations \(\mathcal{I}^w_1, \ldots, \mathcal{I}^w_n\) such that

- **R1** the interpretations are models of \(T\),
- **R2** for any two interpretations \(\mathcal{I}', \mathcal{I}'' \in \{\mathcal{I}^w_1, \ldots, \mathcal{I}^w_n\}\), we have \(\Delta \mathcal{I}' = \Delta \mathcal{I}''\) and \(X \mathcal{I}' = X \mathcal{I}''\) for all \(X \in \text{rig}\),
- **R3** for all \(i \in [1, k]\), \(\mathcal{I}^w_i \models \bigwedge_{q \in X^w_i} q \wedge \bigwedge_{q \notin X^w_i} \neg q\), and
- **R4** for all \(i \in [1, n]\), \(\mathcal{I}^w_i \models \bigwedge_{q \in a(w, i)} q \wedge \bigwedge_{q \notin a(w, i)} \neg q\) and \(\mathcal{I}^w_i \models \mathcal{A}'_i\).
\( \iota \) is now defined by setting for all \( w \in \Omega_K \) and \( j \geq 1 \):

1. if \( j \in [1, n] \) \( \iota(w)_j = I^w_j \), and
2. for all \( j > n \) \( \iota(w)_j = J^w_j \), where \( j \) is such that \( Q_1(w) \cap \{ q_1, \ldots, q_m \} = X^w_j \) by Condition \( Q_6 \).

By Condition \( R_1 \), \( \iota \) respects rigid names. Furthermore, for every \( w = (A^j_i)_{j \in [1,n]} \in \Omega_K \) and \( j \geq 1 \), \( \iota(w)_j \models T \) and if \( j \in [1,n] \) \( \iota(w)_j = A^j_i \). It follows that \( \iota \) is a model of \( K \). It remains to show that \( \iota \) satisfies \( \phi \) at \( i \). We do this by structural induction on \( \phi \) and show for every \( \psi \in T, w \in \Omega_K \) and \( j \geq 1 \), \( \iota(w)_j \models \psi \iff \psi \in Q_j(w) \). Since \( \phi \in Q_i(w) \) for all \( w \in \Omega_K \), this proves that \( \iota, i \models \phi \). We distinguish the cases based on the syntactical shape of \( \psi \).

1. \( \psi \) is a CQ. We have two cases. Either i) \( j \in [1, n] \) and \( \iota(w)_j = T^w_j \), or ii) \( j > n \) and \( \iota(w) = J^w_j \) for some \( j' \in [1, k] \). In the first case, \( \psi(a(w, j)) \iff \psi \) by Condition \( R_4 \) and \( \psi(a(w, j)) \iff \psi \in Q_j(w) \) by Condition \( Q_7 \). Consequently, \( \psi \in Q_j(w) \iff \psi \models \iota, w, j \models \psi \).

In the second case, by Condition \( R_3 \) we have \( \psi \in X^w_j \) \( \iota(w)_j \models \psi \). Furthermore, we have \( Q_j(w) \) \( \iota, w, j \models \psi \) \( \psi \in X^w_j \) by construction, and consequently \( \psi \in Q_j(w) \) \( \psi \in X^w_j \). It follows that \( \psi \in Q_j(w) \iff \psi \models \iota, w, j \models \psi \).

2. \( \psi = \neg \psi' \). By inductive hypothesis, \( \iota, w, j \models \psi \iff \psi \in Q_j(w) \). Consequently, by the semantics of negation and Condition \( S_1 \), \( \iota, w, j \models \psi \iff \psi \models \iota, w, j \models \psi \).

3. \( \psi = \psi_1 \land \psi_2 \). By inductive hypothesis, \( \iota, w, j \models \psi_1 \iff \psi_1 \in Q_j(w) \), and \( \iota, w, j \models \psi_2 \iff \psi_2 \in Q_j(w) \). By the semantics of conjunction and Condition \( S_3 \), we obtain that \( \iota, w, j \models \psi_1 \land \psi_2 \iff \psi_1 \land \psi_2 \models \iota, w, j \models \psi_1 \land \psi_2 \).

4. \( \psi = \bigcirc \psi' \). By Condition \( Q_2 \), \( \bigcirc \psi' \models Q_j(w) \iff \psi' \models Q_{j+1}(w) \). By inductive hypothesis, \( \psi' \models Q_{j+1}(w) \iff \iota, w, j + 1 \models \psi' \). By the semantics of the TPQs, \( \iota, w, j + 1 \models \psi' \iff \iota, w, j \models \bigcirc \psi' \). Consequently, \( \bigcirc \psi' \models Q_j(w) \iff \iota, w, j \models \bigcirc \psi' \). The case where \( \psi = \bigcirc \neg \psi' \) is similar.

5. \( \psi = \psi_1 \mathcal{U} \psi_2 \). Assume \( \psi_1 \mathcal{U} \psi_2 \models Q_j(w) \). By Condition \( Q_4 \), there then exists \( j' \geq j \) s.t. \( \psi_2 \models Q_{j'}(w) \) and \( \psi_1 \models Q_{j'}(w) \). By the inductive hypothesis, we then also have \( \iota, w, j' \models \psi_2 \) and \( \iota, w, k' \models \psi_2 \) for all \( k' \in [j, j'] \), which, by the semantics of the until-operator, implies \( \iota, w, j \models \psi_1 \mathcal{U} \psi_2 \). For the other direction, assume \( \iota, w, j \models \psi_1 \mathcal{U} \psi_2 \). By the semantics of the until-operator, there then exists some \( j' \geq j \) s.t. \( \iota, w, j' \models \psi_2 \) and \( \iota, w, k' \models \psi_1 \) for all \( k' \in [j, j'] \), which by Condition \( Q_4 \) implies that \( \psi_1 \mathcal{U} \psi_2 \models Q_j(w) \). The case where \( \psi = \psi_1 \mathcal{S} \psi_2 \) is similar.

We obtain that for every \( \psi \in T(\phi) \), \( w \in \Omega \) and \( j \geq 1 \), \( \iota, w, j \models \psi \iff \psi \models Q_j(w) \). Since \( \phi \in Q_i(w) \) for all \( w \in \Omega_K \), this implies \( \iota, i \models \phi \). Since \( \iota \) is a model of \( K \), we obtain that \( \phi \) is satisfiable in \( K \) at \( i \). \( \square \)

We can now prove Lemma \( 7 \) from the main text.

**Lemma 7.** \( \phi \) is satisfiable in \( K \) at time point \( i \) iff there exist mappings \( S : \Omega_K \to 2^{2^{|q_1, \ldots, q_m|}} \) and \( a : \Omega_K \times [1, n] \to 2^{2^{|q_1, \ldots, q_m|}} \) and a quasi-model \( (Q_j)_{j \geq 1} \) for \( K \) such that

1. \( \phi \in Q_i(w) \) for all \( w \in \Omega_K \),
2. \( (Q_j)_{j \geq 1} \) is compatible with \( S \) and \( a \),
3. \( S \) is \( r \)-satisfiable wrt. to \( a \), and
4. \( (Q_j)_{j \geq 1} \) is of the form \( Q_1, \ldots, Q_m(Q_{m+1}, \ldots Q_{m+o})^2 \), where \( m \) and \( o \) are both double exponentially bounded in the size of \( K \).
Proof. The direction (⇐) follows directly from Lemma[13] We therefore only have to show the other direction.

Let \( S : \Omega_K \rightarrow 2^{\{q_1, \ldots, q_m\}} \) and \( a : \Omega_K \times [1, n] \rightarrow 2^{\{q_1, \ldots, q_m\}} \) be mappings and \((Q_j)_{j \geq 1}\) be a quasi-model for \( \phi \) compatible with \( S \) and \( a \). First note that there can be at most double-exponentially many distinguishable quasi-states in \((Q_j)_{j \geq 1}\): \( \Omega_K \) contains at most \( 2^{|K|} \) many elements, and for each \( w \in \Omega_K \), \( Q_j(w) \) contains at most \(|\phi|\) elements. We obtain that there are at most \( 2^{2^{K} \cdot |\phi|} \) many different combinations. For indices \( j, k \) s.t. \( Q_j = Q_k \), we define an operation merging of \( Q_j \) and \( Q_j \) in \((Q_j)_{j \geq 1}\), which replaces the quasi-model with \( Q_1, \ldots, Q_i, Q_{j+1}, \ldots \) One can verify that the result of merging in a quasi-model is again a quasi-model: Condition \( Q_1 \) Conditions \( Q_1 \), \( Q_3 \) only consider at most two-subsequent states, and 2) Conditions \( Q_4 \) and \( Q_5 \) are still satisfied in the new quasi-model. Furthermore, since we only remove quasi-states, the resulting quasi-model also still satisfies Conditions \( Q_6 \) and \( Q_7 \) and is thus compatible to \( S \) and \( a \).

Now let \((Q_j)_{j \geq 1}\) be any quasi-model, and let \( m, o \) be two indices s.t. \( Q_m = Q_{m+o} \) and the following condition is satisfied:

\[
Q_j Q_{j+1} (Q_{j+2} \cdots Q_{j+o})^w.
\]

Such a quasi-model always exists by Condition \( Q_4 \). From (\*), it already follows that \( Q_1, \ldots, Q_m (Q_{m+1}, \ldots Q_{m+o})^w \) is also a quasi-model. However, \( m \) and \( o \) might not be double-exponentially bounded in the size of \( K \) and \( \phi \). By the above observation, we may assume that no quasi-state occurs twice before \( Q_m \), since we can always merge any quasi-states that occur more than once, so that \( m \leq 2^{2^{K} \cdot |\phi|} \). To reduce the index of \( Q_{m+o} \), we exhaustively merge any two quasi-states that occur between \( Q_m \) and \( Q_{m+o} \) for which merging does not break Condition (\*). The resulting quasi-state can now be represented as

\[
Q_1 Q_2 (Q_3 \cdots Q_{m+o})^w.
\]

We give a bound on \( o' \). For every \( i, j \in [n+1, n+o] \) s.t. \( Q_i = Q_j \), there must be some \( w \in \Omega_K \), \( \psi_1 \cup \psi_2 \subseteq Q_m(w) \), and \( k \in [i, j] \) s.t. \( \psi_2 \subseteq Q_k(w) \) and \( \psi_2 \not\subseteq Q_l(w) \) for all \( l \in [n, k-1] \), since otherwise \( Q_i \) and \( Q_j \) would have been merged. It follows that every quasi-state is repeated at most \( 2^{2^{K} \cdot |\phi|} \) times, because there are at most \( 2^{2^{K} \cdot |\phi|} \) possible worlds in \( \Omega_K \) and for each \( w \in \Omega_K \), at most \( |\phi| \) queries of the form \( \psi_1 \cup \psi_2 \subseteq Q_m(w) \). Because the number of distinct quasi-states is bounded by \( 2^{2^{K} \cdot |\phi|} \), we obtain \( o' \leq 2^{2^{K} \cdot |\phi| \cdot 2^{2^{K} \cdot |\phi|}} \), that is, \( o' \) is double-exponentially bounded in the size of \( K \) and \( \phi \). It follows that we can transform any quasi-model into a quasi-model of the required form, and thus that a quasi-model exists iff there exists a regular quasi-model which is of the form as in the lemma.

A.3 Deciding r-Satisfiability

In order to decide r-satisfiability, we construct a classical KB \( K_{a, w} = (A_{a, w}, T_{a, w}) \) based on \( w = (A_i)_{i \in [1, n]} \). \( S = \{X_1, \ldots, X_k\} \) and \( a \), where for each non-rigid name \( Y \in (N_C \cup N_R) \setminus N_{rig} \) and \( X_i \in S(w) \), we use a fresh name \( Y_i \). For every \( A(a)/r(a, b) \in A_i \) \( , A_{a, w} \) contains the assertion \( A_i(a)hr^i(a, b) \) if \( A \not\subseteq N_{rig} \) and \( \not\subseteq N_{rig} \), and otherwise the assertion \( A(a)/r(a, b) \). The TBox \( T_{a, w} \) contains for every axiom \( \alpha \) and every \( X_i \in S(w) \) the axiom \( \alpha^i \), which is obtained from \( \alpha \) by replacing every non-rigid name \( Y \) by \( Y_i \). We do a similar transformation for the queries \( q \in X_i \), to obtain queries \( q^i \).

Lemma 14. \( S \) is r-satisfiable wrt. \( a \) and \( w \) iff the following entailment holds.

\[
K_{a, w} \models \neg \left( \bigwedge_{X_i \in S(w)} \left( \bigwedge_{q \in X_i} q^i \land \bigwedge_{q \not\in X_i} \neg q^i \right) \right).
\]
Proof. ($\Rightarrow$) Assume the entailment holds, and let $\mathcal{I}$ be a model of $\mathcal{K}_{a,w}$ s.t.

$$\mathcal{I} = \left( \bigwedge_{X_i \in S(w)} \left( \bigwedge_{q \in X_i} q^i \land \bigwedge_{q \notin X_i} \neg q^i \right) \right).$$

Based on $\mathcal{I}$, we construct the models $\{\mathcal{I}_1, \ldots, \mathcal{I}_n, \mathcal{J}_1, \ldots, \mathcal{J}_k\}$ as required by Definition 5. For $X_j \in S(w)$, we define the interpretation $\mathcal{J}_j$

1. $\Delta^{\mathcal{J}_j} = \Delta^{\mathcal{I}}$,
2. for all $Y \in N_{\text{rig}}$: $X^{\mathcal{J}_i} = X^{\mathcal{I}}$, and
3. for all $Y \in (N_C \cup N_R) \setminus N_{\text{rig}}$: $Y^{\mathcal{J}_i} = (Y^i)^{\mathcal{J}}$.

For $i \in [1, n]$ and $a(w, i) = X_j$, we define $\mathcal{I}_i = \mathcal{J}_j$. It is now standard to verify the conditions in Definition 5.

- **R1** by the construction of the $\mathcal{T}_{a,w}$, every $\mathcal{J}_i$ is a model of $\mathcal{T}$.
- **R2** rigid names are interpreted the same for all interpretations,
- **R3** for all $i \in [1, k]$, $\mathcal{I}_i = \bigwedge_{q \in X_i} q^i \land \bigwedge_{q \notin X_i} \neg q^i$, and consequently, $\mathcal{J}_i = \bigwedge_{q \in X_i} q \land \bigwedge_{q \notin X_i} \neg q$.
- **R4** for $i \in [1, n]$, $\mathcal{I}_i = \bigwedge_{q \in X_i} q \land \bigwedge_{q \notin X_i} \neg q$ for the same reason as in the last case, and since $\mathcal{I} = \alpha_i$ for all $\alpha \in \mathcal{A}_i$, $\mathcal{I}_i = \mathcal{A}_i$.

We obtain that $S$ is r-satisfiable wrt. $a$ and $w$.

($\Rightarrow$) Now for the other direction, assume $S$ is r-satisfiable wrt. $a$ and $w$, and let $\mathcal{I}_1, \ldots, \mathcal{I}_n, \mathcal{J}_1, \ldots, \mathcal{J}_k$ be the interpretations from Definition 5. Note that, since $a(w, i) \in S(w)$ for all $i \in [1, n]$, we may assume without loss of generality that $\{\mathcal{I}_1, \ldots, \mathcal{I}_n\} \subseteq \{\mathcal{J}_1, \ldots, \mathcal{J}_k\}$: if there exists interpretations $\mathcal{I}_1, \ldots, \mathcal{I}_n, \mathcal{J}_1, \ldots, \mathcal{J}_k$ satisfying the conditions in Definition 5 then there also exist such interpretation so that $\{\mathcal{I}_1, \ldots, \mathcal{I}_n\} \subseteq \{\mathcal{J}_1, \ldots, \mathcal{J}_k\}$. We may further assume without loss of generality that for $i \in [1, n]$, $\mathcal{I}_i = \mathcal{J}_i$.

We construct an interpretation $\mathcal{I}$ as follows.

1. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}_i}$,
2. for all $Y \in N_{\text{rig}}$, $Y^{\mathcal{J}} = X^{\mathcal{J}_i}$, and
3. for all $Y \in (N_C \cup N_R) \setminus N_{\text{rig}}$ and $i \in [1, k]$, $(Y^i)^{\mathcal{J}} = Y^{\mathcal{J}_i}$.

Note that by Condition **R2** the specific choice of an interpretation in Step 1 and 2 does not affect the result. We show that $\mathcal{I}$ is a model of $\mathcal{K}_{a,w}$ s.t.

$$\mathcal{I} = \left( \bigwedge_{X_i \in S(w)} \left( \bigwedge_{q \in X_i} q^i \land \bigwedge_{q \notin X_i} \neg q^i \right) \right).$$

By Condition **R1** for every $j \in [1, k]$ and every $\alpha \in \mathcal{T}$, $\mathcal{J}_j = \alpha$, and consequently, by construction of $\mathcal{I}$, one easily sees that $\mathcal{I} = \alpha^i$. It follows that $\mathcal{I}$ is a model of $\mathcal{T}_{a,w}$. By Condition **R4** for every $i \in [1, n]$ and $\alpha \in \mathcal{A}_i$, we have $\mathcal{J}_i = \mathcal{I}_i = \alpha$, which similarly by construction implies $\mathcal{I} = \alpha^i$. It follows that $\mathcal{I}$ is a model of $\mathcal{K}_{a,w}$. It remains to show that for the queries $q \in \{q_1, \ldots, q_m\}$, and for every $i \in [1, k]$, $\mathcal{I} = q^i$ iff $q \in X_i$. Again this directly follows from construction, where we now consider Condition **R3**

\[\square\]
We first sketch the idea to decide query entailment for CQs. We obtain that, in order to use this to decide r-satisfiability, we have to decide the entailment of an.

We next show that the entailment in Lemma 14 can be decided within the desired complexity bounds. By

The main result is given in Theorem 32 in that paper. The main construction underlying the decision

For a query \( q \), denote by \( A_q \) the classical ABox obtained from \( q \) by replacing every variable by a fresh individual name. Clearly, our non-entailment holds if the following non-entailment is satisfied.

We obtain that, in order to use this to decide r-satisfiability, we have to decide the entailment of an exponentially-sized union of linearly sized CQs from an exponentially sized KB. For \( \mathcal{ELH} \) and \( SHQ \) with \( N_{\text{reg}} \neq 0 \), this already gives us the desired complexity bounds, since query entailment is in NP for \( \mathcal{ELH} \) \(^{22}\) and in \( \text{ExpTime} \) for \( SHQ \) \(^{28}\), thus giving us \( \text{NExpTime} \) and \( 2\text{-ExpTime} \) upper bounds immediately. On the other hand, for DLs extending \( \mathcal{ALCI} \) and \( \mathcal{ALCO} \), query entailment is \( 2\text{-ExpTime}-\text{hard} \) \(^{27,30}\), so that we would obtain at least a \( 3\text{-ExpTime} \) upper bound if we just apply these complexity bounds directly. To show that the above entailment can still be decided in \( 2\text{-ExpTime} \) wrt. \( K \) and \( \phi \), we examine the decision procedures presented in \(^{22,21}\) to show that query entailment is in \( 2\text{-ExpTime} \) for \( SHIQ \), \( SHOIQ \) and \( SHOT \), and show that they run in \( 2\text{-ExpTime} \) wrt. \( K \) and \( \phi \) for the above entailment. The crucial property here is that, even though the KB and the UCQ are in \( SHIQ \) \( -ABoxes \), the authors define the set \( \langle \neg X \rangle \) for the above entailment. The crucial property here is that, even though the KB and the UCQ are

\[ K_{\alpha,w} \not\models \bigvee_{X,i \in S(w),q \in X_i} \neg q^i \bigvee_{X,i \in S(w),q \not\in X_i} q^i \]

For a query \( q \), denote by \( A_q \) the classical ABox obtained from \( q \) by replacing every variable by a fresh individual name. Clearly, our non-entailment holds if the following non-entailment is satisfied.

\[ \langle T_{\alpha,w}, A_{\alpha,w} \cup \bigcup_{X,i \in S(w),q \in X_i} A_{q} \rangle \not\models \bigvee_{X,i \in S(w),q \not\in X_i} q^i \]

We show that this procedure runs within the bounds provided by the lemma by direct reference to the results from this publication. The main result is given in Theorem 32 in that paper. The main construction underlying the decision procedure is provided in Definition 21.

We first sketch the idea to decide query entailment for CQs \( q \) from classical KBs \( K = (T, A) \). For a given CQ \( q \), the authors define the set \( \text{trees}_{K}(q) \) which contain rewritings of \( q \) in form of concepts corresponding to all possible tree-shaped matches of \( q \) in models of \( K \), and the set \( \text{ground}_{K}(q) \), which contains ground CQs (that is, CQs that only contain individual names as terms, which may however use complex concepts) corresponding to forest-shaped matches in models of \( K \) in which every root of the forest is matched to an individual in \( K \). The authors define the \( SHIQ \)-TBox \( T_q = \{ \top \subseteq \neg C \mid C \in \text{trees}_{K}(q) \} \), which only has models in which there is a purely-tree shaped match of \( q \). In addition, they define a series of classical \( SHIQ \)-ABoxes \( A_q \), which contain from each ground query \( q' \in \text{ground}_{K}(q) \) at least one assertion \( \neg \alpha \), where \( \alpha \in q' \). If for any such ABox \( A_q \), \( (T \cup T_q, A \cup A_q) \) has a model, then this model is a counter-example for the entailment of \( q \). Note that both \( T_q \) and \( A_q \) are linear in the size of \( \text{trees}_{K}(q) \) and \( \text{ground}_{K}(q) \), and that the number of possible choices for \( A_q \) is exponentially bounded by the size of
The size of each KB is linear in $n$. Relevant for us are the bounds the authors give on their size: by [22] [Lemma 20], the size of trees$_K(q)$ is polynomially bounded in the size of $K$ and exponentially bounded in the size of $q$, and so is the size of ground$_K(q)$.

Now to handle UCQs $q_1 \lor \ldots \lor q_n$, the same procedure is used based on the unions over all sets trees$_K(q_i)$ and ground$_K(q_i)$, $i \in [1, n]$ [22] [Definition 21]. Clearly, if the number of CQs in the UCQ is exponentially bounded by a natural number $n$, and the size of each CQ is polynomially bounded in $n$, then the size of these unions are exponentially bounded in $n$, and there are at most double exponentially many KBs that have to be tested for satisfiability. As a consequence, if the KB and the UCQ are of the shape as in the Lemma, then entailment of the UCQ can be decided in time double exponential in $n$.

**Lemma 16.** For a given natural number $n$, entailment of disjunctions over CQs from SHOQ-KBs can be decided in time double exponential in $n$ if

- the size of the KB is at most exponential in $n$,
- the number of CQs in the disjunction is at most exponential in $n$, and
- the size of each CQ is polynomial in $n$.

**Proof.** The result follows from a close analysis of the 2-EXPTIME decision procedure for entailment of UCQs from SHOQ KBs presented in [21]. The decision procedure follows a similar idea than the one presented in [22], and also apply to the extension $SHOQ^1$ of $SHOQ$ with role conjunctions. Specifically, to decide entailment of a CQ, a set $\text{con}_K(q)$ of CQs of the form $C_1(x_1) \lor \ldots \lor C_n(x_n)$ is defined, where each $C_i$, $i \in [1, n]$ is a $SHOQ^1$-concept and $x_i \neq x_j$ for $i \neq j \in [1, n]$. For UCQs $q$, $\text{con}_K(q)$ is defined as the union of all sets $\text{con}_K(q'_i)$ where $q' \in q$. Intuitively, each $\text{con}_K(q)$ corresponds to a forest-shaped match in some model of $K$, and every such match is represented by some CQ in $\text{con}_K(q)$. A consequence of this is that $K \models \bigvee_{q_i \in \text{con}_K(q)} q_i$ iff $K \models q$ [21] Theorem 6]. The size bounds on $\text{con}_K(q)$ are given in [21] Lemma 7]: its size is at most i) polynomial in the size of $K$ and ii) exponential in the size of $q$. For a UCQ whose number of CQs is exponentially bounded in $n$ and the size of each CQ is polynomially bounded in $n$, from the fact that $\text{con}_K(q)$ corresponds to the union of all sets $\text{con}_K(q'_i)$ where $q'_i \in q$, it follows that $\text{con}_K(q)$ is exponential in both $n$ and the size of $K$.

Now to decide $K \models \bigvee_{q_i \in \text{con}_K(q)} q_i$, we use the fact that the atoms in each query in $\text{con}_K(q)$ are variable-disjoint, and built a sequence of reduction KBs. Each reduction KB is obtained from selecting from each CQ in $q_i \text{con}_K(q)$ one atom $C_i(x_i)$, and adding $\top \subseteq \neg C_i$ to $K$. If one of them is unsatisfiable, $K \not \models \bigvee_{q_i \in \text{con}_K(q)} q_i$ and $K \not \models q$.

The size of each KB is linear in $\text{con}_K(q)$, and there are exponentially many possible choices wrt. the size of $\text{con}_K(q)$. If $q$ is shaped as in the lemma, this amounts to a number of satisfiability tests that is double exponential in $n$, where each KB is exponential in the size of $n$. As shown in the remainder of [21], each such satisfiability test can be performed in time exponential to the size of the KB, so that we obtain that the overall decision procedure runs in time double exponential in $n$.

**Lemma 17.** For a given natural number $n$, entailment of disjunctions over CQs from SHOIQ-KBs can be decided in time double exponential in $n$ if

- the size of the KB is at most exponential in $n$,
- the number of CQs in the disjunction is at most exponential in $n$, and
- the size of each CQ is polynomial in $n$.
Proof. where we inspect the procedure presented in [11] deciding entailment of regular path queries (RPQs) in \( \mathcal{ZOI} \)-KBs. Regular path queries are a generalisation and UCQs, and \( \mathcal{ZOI} \) is an extension of \( \mathcal{SHOI} \), so that this procedure can also be used to decide UCQ entailment for \( \mathcal{SHOI} \)-KBs. The authors in [11] reduce UCQ-entailment to the emptiness problem for one-way non-deterministic parity tree automata (1NPAs). Specifically, given a \( \mathcal{ZOI} \) KB \( \mathcal{K} \) and a RPQ \( q \), they construct a 1NPA \( A_{\mathcal{K} \models q} \) which accepts the empty language iff \( \mathcal{K} \models q \). We first give an overview over the main ideas, before we argue why this approach can decide bounded CQs as in the Lemma in double-exponential time. Specifically, the authors exploit the fact that \( \mathcal{ZOI} \) has the forest model property for query-entailment, that is, query entailment from \( \mathcal{ZOI} \)-KBs \( \mathcal{K} \) can be completely characterized by restricting to models of \( \mathcal{K} \) that can be mapped to a labeled forest where every node represents a domain element, edges correspond to role-connections and the roots of the forest to the named individuals, and every role-connection that does not go to a named individual has a corresponding edge in the forest. To characterise these forest models by means of automata, they represent forest-interpretations directly in labeled trees of fixed branching degree, where the branching degree is bounded by the KB and nodes are labeled with sets of individual, concept and role names. Here, the role names refer to the incoming edge, and individual names only occur on the direct successors of the root ([11] Definition 3.7 and beginning of Section 4). They then construct a fully enriched automaton (FEA) with a polynomial number of states and a constant index, which they step-wise translate into an INPA of \( A_{\mathcal{K}} \) accepting exactly those labeled trees that correspond to a forest-shaped model of \( \mathcal{K} \). \( A_{\mathcal{K}} \) has a double-exponential number of states and a constant index.

To capture query entailment, they define another INPA \( A_{\neg q} \) which accepts exactly the labelled trees that correspond to a forest-shaped interpretation in which the query \( q \) is not entailed, and build the intersection of the automata \( A_{\mathcal{K}} \) and \( A_{\neg q} \), which is the final INPA \( A_{\mathcal{K} \models q} \). If \( A_{\mathcal{K} \models q} \) is empty, that is, the language of accepted trees is empty, there cannot be a forest-shaped model of \( \mathcal{K} \) in which \( q \) is not entailed, and correspondingly, \( \mathcal{K} \models q \).

While [11] only briefly sketch the automaton \( A_{\neg q} \), a construction of an automaton accepting the same language is described in detail in [12]: this automaton has an exponential number of states while its index is exponential. The relevant construction in [12] Section 5] can be easily adapted so that number of states and index depend only on the size of the query, as the only relevant factors are the variables, concept and role names occurring in it.

It is not hard to see that for a UCQ \( q = q_1 \lor \ldots \lor q_m \) the automaton \( A_{\neg q} \) is equivalent to the intersection of the automata \( A_{\neg q_1}, \ldots, A_{\neg q_m} \), as the disjuncts can be tested independently on a given interpretation. Now the number of states and the index of the intersection of two INPAs is determined as follows [12 Proposition 2.15]: Let \( Q(A) \) denote the states of \( A \) and \( \text{ind}(A) \) its index, then

\[
\text{ind}(A_1 \cap A_2) = O(f(A_1, A_2))
\]

and

\[
|Q(A_1 \cap A_2)| \leq 2^{O(f(A_1, A_2)^2)} \cdot f(A_1, A_2) \cdot |Q(A_1)| \cdot |Q(A_2)|,
\]

where \( f(A_1, A_2) = A_1 + A_2 + 1 \). Let \( A'_{\neg q} \equiv A_{\neg q} \) denote the intersection of \( A_{q_1}, \ldots, A_{q_m} \). We obtain that

\[
\text{ind}(A'_{\neg q}) = O \left( \sum_{1 \leq i \leq m} \text{ind}(A_{\neg q_i}) + 1 \right),
\]

and

\[
|Q(A'_{\neg q})| \leq 2^{O \left( \left( \sum_{1 \leq i \leq n} \text{ind}(A_{\neg q_i}) + 1 \right)^2 \right)} \cdot \left( \sum_{1 \leq i \leq n} \text{ind}(A_{\neg q_i}) + 1 \right) \cdot \Pi_{1 \leq i \leq n} |Q(A_{\neg q_i})|.
\]

The authors only explicitly spell out the size of the two-way alternating parity tree (2APA) which they construct in the second-last step of their transformation and then translate to the INPA. The number of states of the 2APA is polynomially bounded in the size of \( \mathcal{K} \) and has a constant index. However, according to [12 Proposition 2.12], the transformation comes with an exponential blow-up in the number of states, while it keeps the index. Therefore, the final INPA \( A_{\mathcal{K}} \) constructed here has a double-exponential number of states and a constant index.
Since for each \( A_{\neg q} \), the \( \text{ind}(A_{\neg q}) \) is single exponential in the size of \( q \), and \( |Q(A_{\neg q})| \) is double exponential in the size of \( q \), we obtain that, provided that \( q \) is shaped as in the lemma, \( \text{ind}(A'_{\neg \neg q}) \) is single exponential in \( n \) and \( |Q(A'_{\neg \neg q})| \) is double exponential in \( n \), and the same holds for \( A'_{K \neg q} = A_K \cap A'_{\neg q} \).

Emptiness of 1NDAs \( A \) can be decided in time \( O(|Q(A)|^{\text{ind}(A)}) \), which is double exponential in \( n \) for \( A'_{K \neg q} \). We obtain that for \( SHOL \)-KBs \( K \), entailment of UCQs \( q \) can be decided in time double exponential in \( n \), provided that the size of \( K \) and the number of CQs in \( q \) is exponentially bounded in \( n \) and the size of each CQ is polynomially bounded in \( n \).

In connection with Lemmata 14, Lemmata 15 and 17 now allow to prove almost all cases in Lemma 6. For the only remaining case where \( L = SHQ \) and \( N_{\text{Reg}} = \emptyset \), we refer to [3, Section 6], where \( r \)-satisfiability is considered in the context of TQ-satisfiability in non-probabilistic, temporal KBs.

Lemma 6. Wrt. the size of \( K \) and \( \phi \), r-satisfiability for \( L \)-TPKBs can be decided in

1. \( \text{NExpTime} \) for \( L = \mathcal{ELH} \).
2. \( \text{NExpTime} \) for \( L = SHQ \) if \( N_{\text{Reg}} = \emptyset \).
3. \( 2 \text{-ExpTime} \) for \( L \in \{ \text{SHIQ}, \text{SHOQ}, \text{SHOT} \} \), and
4. it is decidable for \( L = \text{ALCHOIQ} \).

A.4 Complexity Upper Bounds

We are now ready to prove the complexity upper bound.

Lemma 18. Given the mappings \( S : \Omega_K \rightarrow 2^{2^{(q_1, \ldots, q_m)}} \) and \( a : \Omega_K \rightarrow [1, n] \rightarrow 2^{\{q_1, \ldots, q_m\}} \) s.t. \( S \) is \( r \)-satisfiable wrt. \( a \), it can be decided in \( \text{ExpSpace} \) whether there exists a quasi-model for \( \phi \) in \( K \) wrt. \( S \) and \( a \).

Proof. By Lemma 7 there exists a quasi-model \( Q_1, \ldots \) as in the lemma iff there exists a periodic quasi-model of the form

\[
Q_1, \ldots, Q_m (Q_{m+1}, \ldots, Q_{m+o})^\omega,
\]

where both \( m \) and \( o \) are double-exponentially bounded.

To verify the existence of such a quasi-model in (non-deterministic) exponential space, we proceed as follows. We first guess the numbers \( m \) and \( o \), which both require at most exponentially many bits in binary representation. We then guess the quasi-states \( Q_i \) one after the other, keeping always two proceeding quasi-states in memory, and verify that they satisfy the Conditions Q1 Q3 and additionally that \( \psi_1 U \psi_2 \in Q_i(w) \) iff \( \psi_2 \in Q_i(w) \) or \( \psi_1 U \psi_2 \in Q_{i+1}(w) \), and similarly for queries of the form \( \psi_1 S \psi_2 \).

To verify that each \( S/U \)-formula is eventually satisfied, we keep a set of those queries for each possible world that have not been satisfied yet, which we update at each time point. In the same manner, we check whether the negation of a \( S/U \)-formula is satisfied. After we guessed the quasi-state \( Q_{m+1} \), we store this quasi-state in memory, as well as all \( S/U \)-queries that still have to be satisfied at this point. We then proceed until \( Q_{m+o+1} \), and verify that all \( S/U \)-queries from \( Q_{m+1} \) have been satisfied in the meanwhile, and that \( Q_{m+o+1} = Q_{m+1} \). Since \( \text{NExpSpace} = \text{ExpSpace} \), the above procedure decides existence of a quasi-model in exponential space.

Theorem 8. The complexity upper bounds for general TPQs in Figure 1 hold.

Proof. Entailment of a query \( \phi \) corresponds to non-satisfiability of the query \( \neg \phi \), and both complexity classes \( \text{ExpSpace} \) and \( 2 \text{-ExpTime} \) are closed under complement. To decide satisfiability of a TPQ \( \phi \) with negations in a TPKB \( K \), we proceed as follows. We iterate over the (double exponentially many)
different possible mappings S and α, each of which require only exponential space. For each pair S, α, we first verify whether S is r-satisfiable wrt. α, and then whether there exists a quasi-model that is compatible with S and α. By Lemma 18, the last test takes exponential space in each iteration. The only step that differs depending on the DL in question and which rigid names we allow is verifying r-satisfiability, which by Lemma 6 can be decided in NEXPTime (and therefore in ExpSpace) for L = EL, as well as for L = SHIQ provided that NRig = ∅. For L ∈ {SHIQ, SHOIQ, SHOT}, it can be decided in 2-ExpTime. We thus obtain the required complexity bounds. For L = ALCCHIQ, it suffices to use that r-satisfiability is decidable for this logic.

A.5 Positive Temporal Probabilistic Queries

We first establish the complexity bounds for the non-probabilistic case, that is, entailment of temporal queries (TQs), which are TPQs without probability operators, from temporal knowledge bases (TKBs), which are TPKBs without probabilities different from 1.

Lemma 19. Entailment of positive TQs is in P wrt. data complexity and NP complete wrt. combined complexity, even if NCriq ≠ ∅ and NRig ≠ ∅.

Proof. [7] show that query entailment is in P data complexity, even if NCriq ≠ ∅. The only remaining case is the combined complexity for EL TKBs. To establish the combined complexity, we describe an NP procedure for a given positive TQ φ and EL TKB (T, (Ai)ε[1,n]). For X ∈ NC ∪ NR, denote by X(i) the name X if X ∈ NR, and a fresh name X′ if X ∉ NR, and for a given axiom/assertion/query α, denote by α(i) the result of replacing every name X in α by X(i). Define an atemporal KB K′ = {T′, A′} based on the TKB K = (T, (Ai)ε[1,n]) by T′ = {α(i) | α ∈ T, i ∈ [1, n + 1]} and A′ = {α(i) | α ∈ Ai, i ∈ [1, n + 1]}, K′ is polynomial in K, and one can show that for any axiom/assertion/CQ α and i ∈ [1, n + 1], we have K, i ⊨ α iff K′ ⊨ α(i) [8].

In order to decide entailment of a TQ φ, we guess a certificate that assigns to each pair (i, ψ) of a time point i ∈ [1, n + nt] and a CQ ψ occurring in φ a truth value, and, in case true is assigned to such a pair (i, ψ), a certificate for the entailment of ψ at i (such a certificate exists since entailment of CQs is in NP wrt. combined complexity). For any time point after n, the entailment of a CQ solely depends on the rigid names. Therefore, for every CQ q in φ, if K, n + 1 ⊨ q, then K, n + i ⊨ q for all i > 1. Based on the guessed truth-assignment of CQs, we can now evaluate the entailment of φ as in the propositional case, which for LTL-formulae without negation symbols can be done in P [9]. As this certificate can be guessed and verified in non-deterministic polynomial time, we obtain an NP-upper bound. □

The proof of Theorem 10 further depends on the following lemma, which limits the time points we have to consider explicitly.

Lemma 20. Let φ be a TPQ, K = (T, (Ai)ε[1,n]) a TKB and nt be the maximal nesting depth of temporal operators in φ. Then, for every i > n + nt, K, i ⊨ φ iff K, n + n + 1 ⊨ φ.

Proof. We do the proof by structural induction on φ, and distinguish the cases based on the structure of φ.

1. If φ is a CQ, note that the only way in which K restricts its models for time points after n is via its rigid names. Therefore, we have for all i > n, K, i ⊨ φ iff K, n + 1 ⊨ φ.

2. If φ is of one of the forms ψ1 ∧ ψ2 and ψ1 ∨ ψ2, the hypothesis follows by direct application of the inductive hypothesis.
3. If $\phi$ is of the form $\mathcal{O} \neg \psi$, we have by inductive hypothesis that for all $i > n + n_t - 1$, $\mathcal{K}, n + n_t + 1 \models \psi$ iff $\mathcal{K}, i \models \psi$ iff $\mathcal{K}, i + 1 \models \mathcal{O} \neg \psi$ iff $\mathcal{K}, j \models \phi$ for all $j > n + n_t$.

4. If $\phi$ is of one of the forms $\mathcal{O} \psi, \Diamond \psi, \Diamond \mathcal{O} \psi, \Box \psi, \Box \mathcal{O} \psi, \psi_1 \cup \psi_2$ or $\psi_1 \mathcal{S} \psi_2$, we note that by inductive hypothesis, for all $i > n + n_t - 1$, $\mathcal{K}, i \models \psi_1$ (or $\psi_2$) iff $\mathcal{K}, n + n_t \models \psi_1$, which implies $\mathcal{K}, i \models \phi$ iff $\mathcal{K}, n + n_t + 1 \models \psi$ for all $i > n + n_t - 1$, and consequently also $\mathcal{K}, i \models \phi$ iff $\mathcal{K}, n + n_t + 1 \models \phi$ for all $i > n + n_1$.

\[ \square \]

We can now provide the upper bounds stated in Theorem 10. A central technique used for this is to flatten TPQs using an abstraction of the probability expressions $P_{\geq p}(\psi)$ occurring in the query. We identify each such expression with the assertion $A_{p,\psi}(a)$, where $A_{p,\psi}$ is fresh, which we add to the ipABox $A_i$ once we established that $P_{\geq p}(\psi)$ is entailed at $i$. To capture this abstraction in a given TPQ $\psi$, we denote by $\psi_f$ the result of replacing every outermost sub-query in $\psi$ of the form $P_{\geq p}(\psi)$ with $\exists x. A_{p,\psi}(x)$.

**Lemma 21.** Entailment of TPQs from $\mathcal{EL}$ and $\mathcal{DL}$-Lite-TPKBs is in PP wrt. data complexity, even if $\mathcal{N}_{\text{orig}} \neq \emptyset$. It is in $\mathcal{PP} \mathcal{NP}$ wrt. combined complexity if the nesting depth of probability-operators in the query is bounded, and otherwise in $\mathcal{P}^{\mathcal{PP}}$.

**Proof.** Before we consider nested probability operators, we consider the basic case of simple TPQs of the form $P_{\geq p}(\phi)$, where $\phi$ does not contain any probability operators. Entailment of such a TPQ can be decided by checking for which possible world $w \in \Omega_{\mathcal{K}}$, $w, i \models \phi$, and then summing the probabilities of these worlds. This can be implemented by a probabilistic Turing machine (which uses an NP-oracle in the case of the combined complexities), which constructs a single possible world $w = (A'_i)_{i \in [1, n]}$ on each branch, while taking care that the probabilities of the possible worlds are reflected by the probabilities in the Turing machine. For each $i \in [1, n]$ and $\alpha : p \in A_i$, the machine adds $\alpha$ to $A'_i$ on $b_1$ succeeding branches, and does not add $\alpha$ to $A'_i$ on $b_2$ succeeding branches, where $\frac{b_1}{b_1 + b_2} = p$. After all axioms are processed, accept if $\langle T, (A'_i)_{i \in [1, n]} \rangle, i \models \phi$, which can be decided in $\mathcal{P}$ data complexity and $\mathcal{NP}$ combined complexity. By adding further dummy states to the Turing machine, we can ensure that the machine accepts at least half of its computation paths iff $\mathcal{K} \models P_{\geq p}(\phi)$, so that entailment of the simple TPQ $\phi$ is decided in $\mathcal{P}$ data complexity and $\mathcal{PP} \mathcal{NP}$ combined complexity.

To decide entailment of TPQS that contain several probability operators, we proceed in $k$ rounds, where $k$ is the maximal nesting depth of probability operators in $\phi$, and test in each round for the entailment of probabilistic sub-queries at different time points. Let $n_t$ denote the maximal nesting depth of temporal operators in $\phi$. It can be shown that we have to consider only the first $n + n_t$ time points. In each round $r \in [1, k]$, we iterate over all subformulae in $\phi$ that are of the form $P_{\geq p}(\psi)$, where $\psi$ contains at most $r - 1$ nestings of probability operators, and over all time points $i \in [1, n + n_t + 1]$, and decide whether $\mathcal{K}, i \models P_{\geq p}(\psi_f)$. If $\mathcal{K}, i \models P_{\geq p}(\psi_f)$, we add $A_{p,\psi}(a)$ to $A_i$. In the last round, we processed all probability operators, and decide whether $\mathcal{K} \models P_{\geq 1}(\phi_k)$. Provided the nesting depth of probability operators is bounded, (as is always the case for data complexity), we can now use the fact that $\mathcal{P}$ (and therefore also $\mathcal{PP} \mathcal{NP}$) is closed under $k$-round polynomial truth table reductions [19]. These are defined as a sequence of $k$ sets of polynomially many polynomial truth-table reductions, where $k$ is a constant, and each truth-table reduction only depends on the input and the results of previous rounds. If the nesting-depth of probability operators is bounded, the above procedure can be described by such a reduction, and we obtain the $\mathcal{P}$ and $\mathcal{PP} \mathcal{NP}$ upper bounds. Regarding the combined complexity with unbounded nesting of probability operators, we note that the above procedure can be implemented by a polynomial Turing machine that decides entailment of simple TPQs using a $\mathcal{PP} \mathcal{NP}$ oracle, so that we obtain a $\mathcal{P}^{\mathcal{PP} \mathcal{NP}}$ upper bound. Now, using Toda’s result that $\mathcal{P}^{\mathcal{PH}} \subseteq \mathcal{PP} \mathcal{NP}$, we can internalise all calls to the $\mathcal{PP} \mathcal{NP}$ oracle in a $\mathcal{PP}$ machine, so that we obtain a $\mathcal{P}^{\mathcal{PP}}$ upper bound for the combined complexity without bound on the nesting-depth of probability operators.

\[ \square \]
For the data complexity, our upper bound is matched by PP-hardness of the atemporal case [24]. We could not find a lower bound for the combined complexity in the literature for our precise setting (ipABoxes or tuple-independent databases). We therefore provide a proof for it here.

**Lemma 22.** Entailment of TPQs from TPKBs is \( \text{PP}^{\text{NP}} \)-hard.

**Proof.** We only need to provide a lower bound for the combined complexity. We do the proof by reduction of the \( \text{PP}^{\text{NP}} \) complete problem \( \exists \text{CNF3} \): given a QBF-formula of the form \( \phi = \exists x_1, \ldots, x_n, \phi' \), where \( \phi \) is a CNF3-formula over the variables \( \{x_1, \ldots, x_n, y_1, \ldots, y_m\} \) with clauses \( \{c_1, \ldots, c_o\} \), decide whether at least half of assignments of truth values to the variables \( y_1, \ldots, y_m \) make \( \phi \) true [35]. As it turns out, we only need a single ipABox for this. The ipABox contains for every variable \( x_i, i \in [1, n] \) the assertions \( B(x_i) \) and \( \overline{B}(x_i) \), and for every variable \( y_i, i \in [1, m] \) the assertions \( B_i^+ (y_i) \): 0.5 and \( B_i^- (y_i) \): 0.5. Intuitively, \( B_i^+ (y_i) \) is entailed in a possible world that corresponds to an assignment of true to the variable \( y_i \), while \( B_i^- (y_i) \) is entailed in a possible world that corresponds to an assignment of false to the variable \( y_i \). Since all probabilities are independent, we will have worlds that correspond to "invalid variable assignments", in the sense that they either do not assign a truth value to every variable, or multiple truth values. We will take care of this later. We use the TBox axioms \( B_i^+ \subseteq B_i, B_i^- \subseteq B_i, B_i^+ \subseteq B^+, B_i^- \subseteq B^- \), \( B^+ \subseteq B \) and \( B^- \subseteq B \) to abstract away from the specific assignment if needed.

For every literal \( l \), denote by \( v(l) \) the variable in \( l \). For every clause \( c_j = l_1 \lor l_2 \lor l_3, c \in [1, o] \), and truth valuation \( \pi \) that makes \( c_j \) true, add the assertions

\[
M(c_j, \pi), M(\pi, l_1'), M(\pi, l_2'), M(\pi, l_3'),
\]

where for \( i \in [1, 3], l_i' = v(l_i) \) if \( \pi(l_i) = \text{true} \), and \( l_i' = \overline{v}(l_i) \) if \( \pi(l_i) = \text{false} \). As last assertion, we add \( H(a): 0.5 \), which only serves the purpose of being satisfied in at least half of the possible worlds.

Our CQ is now composed of three queries \( q_1, q_2 \) and \( q_3 \) defined next. The query

\[
q_1 = \exists y_1, \ldots, y_m : B_1(y_1) \ldots B_n(y_m)
\]

is entailed in every possible world which assigns a truth value to each variable \( y_i, i \in [1, m] \). The query

\[
q_2 = \exists y : B^+(y) \land B^-(y)
\]

is entailed in the possible worlds that assign two truth values to some variable \( y \). Finally, the query

\[
q_3 = \exists x_1, \ldots, x_n, y_1, \ldots, y_m, z_1, \ldots, z_o : \bigwedge_{i \in [1, o]} \tau(c_i),
\]

where for \( c_i = l_1 \lor l_2 \lor l_3, \)

\[
\tau(c_i) = M(c_i, z_i), M(z_i, v(l_1)), M(z_i, v(l_2)), M(z_i, v(l_3))
\]

\[
\land \bigwedge_{i \in [1, 3]} B(v(l_i))
\]

is satisfied in all possible worlds that correspond to an assignment that make \( \phi \) true. \( q_3 \) can only be entailed in a possible world in which \( q_1 \) is also entailed (otherwise, we lack variables for some of the clauses). The query \( (q_1 \land q_2) \lor (H(c) \land q_2) \) is entailed in (1) all possible worlds that correspond to an assignment that is complete but assigns to at least one variable two values and (2) half of the possible worlds that correspond to assignments that are both incomplete and assign two values to a variable. Due to symmetry, this query is thus entailed in exactly half of those possible worlds that do not correspond to a valid variable assignment. Consequently, the query

\[
q = (q_1 \land q_2) \lor (H(c) \land q_2) \lor q_3
\]

is entailed in more than half of all possible worlds iff \( \phi \) is satisfied for more than half of its valid assignments, so that \( P_{\geq 0.5}(q) \) is entailed iff \( \phi \) is satisfied by at least half of the assignments. Note furthermore that both the TPKB and the TPQ are polynomial in the input, so that we obtain the \( \text{PP}^{\text{NP}} \) lower bound.

\( \square \)