TECHNISCHE UNIVERSITÄT DRESDEN

Technische Universität Dresden
Institute for Theoretical Computer Science
Chair for Automata Theory

# LTCS–Report

## Using Ontologies to Query Probabilistic Numerical Data (Extended Version)

Franz Baader Patrick Koopmann Anni-Yasmin Turhan

LTCS-Report 17-05

This is an extended version of the article in: Proceedings of the 11th International Symposium on Frontiers of Combining Systems. This version has been revised based on the comments of the reviewers.

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

# Using Ontologies to Query Probabilistic Numerical Data (Extended Version)*

Franz Baader Patrick Koopmann Anni-Yasmin Turhan

July 4, 2017

**Abstract**

We consider ontology-based query answering in a setting where some of the data are numerical and of a probabilistic nature, such as data obtained from uncertain sensor readings. The uncertainty for such numerical values can be more precisely represented by continuous probability distributions than by discrete probabilities for numerical facts concerning exact values. For this reason, we extend existing approaches using discrete probability distributions over facts by continuous probability distributions over numerical values. We determine the exact (data and combined) complexity of query answering in extensions of the well-known description logics $\mathcal{EL}$ and $\mathcal{ALC}$ with numerical comparison operators in this probabilistic setting.

## Contents



*Supported by the DFG within the collaborative research center SFB 912 (HAEC) and the research unit FOR 1513 (HYBRIS).

# 1 Introduction

*Ontology-based query answering (OBQA)* has recently attracted considerable attention since it dispenses with the closed world assumption of classical query answering in databases and thus can deal with incomplete data. In addition, background information stated in an appropriate ontology can be used to deduce more answers. OBQA is usually investigated in a setting where queries are (unions of) conjunctive queries and ontologies are expressed using an appropriate Description Logic (DL). Depending on the expressiveness of the DL, the complexity of query answering may vary considerably, starting with data complexity (i.e., complexity measured in the size of the data only) of $\mathrm{AC}^0$ for members of the DL-Lite family [9, 2] to P for DLs of the $\mathcal{EL}$ family [31], all the way up to intractable data complexity for expressive DLs such as $\mathcal{ALC}$ and beyond [18].

In many application scenarios for OBQA, however, querying just symbolic data is not sufficient. One also wants to be able to query numerical data. For example, in a health or fitness monitoring application, one may want to use concepts from a medical ontology such as SNOMED CT [17] or Galen [32] to express information about the health status of a patient, but also needs to store and refer to numerical values such as the blood pressure or heart rate of this patient. As an example, let us consider hypertension management using a smartphone app [24]. What constitutes dangerously high blood pressure (HBP) depends on the measured values of the diastolic pressure, but also on other factors. For example, if a patient suffers from diabetes, a diastolic blood pressure above 85 may already be classified as too high, whereas under normal circumstances it is only considered to be too high above 90. This could, for example, be modelled as follows by an ontology:

$$\exists \mathsf{diastolicBloodPressure}.{>_{90}} \sqsubseteq \mathsf{PatientWithHBP} \tag{1}$$

$$\exists \mathsf{finding}.\mathsf{Diabetes} \sqcap \exists \mathsf{diastolicBloodPressure}.{>_{85}} \sqsubseteq \mathsf{PatientWithHBP} \tag{2}$$

Note that we have used a DL with concrete domains [6] to refer to numerical values and predicates on these values within concepts. While there has been quite some work on traditional reasoning (satisfiability, subsumption, instance) in DLs with concrete domains [27], there is scant work on OBQA for such DLs. To the best of our knowledge, the only work in this direction considers concrete domain extensions of members of the DL-Lite family [3, 34, 4, 20]. In

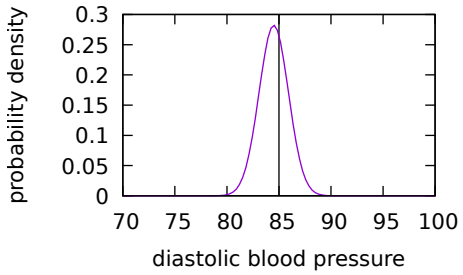


Figure 1: Measured blood pressure as normal distribution.

contrast, we consider concrete domain extensions of $\mathcal{EL}$ and $\mathcal{ALC}$ and determine the (combined and data) complexity of query answering.

However, the main difference to previous work is that we do not assume the numerical values in the data to be exact. In fact, a value of 84.5 for the diastolic pressure given by a blood pressure sensor does not really mean that the pressure is precisely 84.5, but rather that it is around 84.5. The actual value follows a probability distribution—for example a normal distribution with expected value 84.5 and a variance of 2 as shown in Figure 1—which is determined by the measured value and some known variance that is a characteristic of the employed sensor. We can represent this in the knowledge base for example as follows:

$$\mathsf{finding}(\mathsf{otto},\mathsf{f1}) \qquad \mathsf{Diabetes}(\mathsf{f1}) \qquad \mathsf{diastolicBloodPressure}(\mathsf{otto}) \sim \mathsf{norm}(84.5, 2)$$

From this information, we can derive that the minimal probability for the patient Otto to have high blood pressure is slightly above 36%, which might be enough to issue a warning. In contrast, if instead of using a probability distribution we had asserted 84.5 as the exact value for Otto's diastolic blood pressure, we could not have inferred that Otto is in any danger.

Continuous probability distributions as used in this example also emerge in other potential applications of OBQA such as in robotics [37], tracking of object positions in video analytics [39], and mobile applications using probabilistic sensor data [15], to name a few. The interest in continuous probability distributions is also reflected in the development of database systems that support these [36].

In addition to using continuous probability distributions for sensor values, we also consider discrete probability distributions for facts. For example, it might be that the finding f1 for Otto is diabetes only with a certain probability. While OBQA for probabilistic data with discrete probability distributions has been considered before for DL-Lite and $\mathcal{EL}$ without concrete domains [14, 22, 12], as well as for datalog [11], OBQA for probabilistic data with both discrete and continuous probability distributions is investigated here for the first time. A rather expressive combination we consider is the DL $\mathcal{ALC}$ extended with a concrete domain in which real numbers can be compared using the (binary) predicates $>$ and $=$. A less expressive combination we consider is the DL $\mathcal{EL}$ extended with a concrete domain in which real numbers can be compared with a fixed number using the (unary) predicates $>_n$ for $n \in \mathbb{R}$. Since OBQA for classical knowledge bases (i.e., without probabilities) in these two DLs has not been investigated before, we first determine their (data and combined) complexity of query answering. When considering probabilistic KBs with continuous probability distributions (modelled as real-valued functions), the resulting probabilities may be numbers without a finite representation. To overcome this problem, we define probabilistic query entailment with respect to a given precision parameter. To allow a reasonable complexity analysis, we define a set of feasibility conditions for probability distributions, based on the complexity theory of real functions [23], which capture most typical probability distributions that appear in practical applications. For probabilistic KBs

that satisfy these conditions, we give tight bounds on the complexity of probabilistic query answering w.r.t data and combined complexity for all considered DLs.

Detailed proofs for all results can be found in the appendix.

# 2 Description Logics with Numerical Domains

We recall basic DLs with concrete domains, as introduced in [6], and give complexity results for classical query answering.

A *concrete domain* is a tuple $\mathcal{D} = (\Delta_\mathcal{D}, \Phi_\mathcal{D})$, where $\Delta_\mathcal{D}$ contains objects of the domain, and $\Phi_\mathcal{D}$ contains predicates $P_n$ with associated arity $n$ and extension $P_n^\mathcal{D} \subseteq \Delta_\mathcal{D}^n$. Let $N_c$, $N_r$, $N_{cF}$ and $N_i$ be pair-wise disjoint sets of *names* for *concepts*, *roles*, *concrete features* and *individuals*, respectively. Let $N_{aF} \subseteq N_r$ be a set of *abstract feature names*. Concrete features are partial functions that map individuals to a value in the concrete domain. Abstract features are functional roles and their use in *feature paths* does not harm decidability [26]. A *feature path* is an expression of the form $u = s_1 s_2 \dots s_n g$, where $s_i \in N_{aF}$, $1 \leq i \leq n$, and $g \in N_{cF}$. $\mathcal{ALC}(\mathcal{D})$ concepts are defined as follows, where $A \in N_c$, $s \in N_r$, $u$ and $u'$ are feature paths, $P_n \in \Phi_\mathcal{D}$ is a predicate of arity $n$, and $C_1$ and $C_2$ are $\mathcal{ALC}(\mathcal{D})$ concepts:

$$C \;:=\; \top \;\mid\; A \mid \neg C_1 \;\mid\; C_1 \sqcap C_2 \;\mid\; \exists s.C_1 \;\mid\; \exists(u_1, \dots, u_n).P_n \;\mid\; u{\uparrow}.$$

Additional concepts are defined as abbreviations: $C_1 \sqcup C_2 = \neg(\neg C_1 \sqcap \neg C_2)$, $\forall s.C = \neg\exists s.\neg C$, and $\bot = \neg\top$. If a concept uses only the constructors $\top$, $A$, $C_1 \sqcap C_2$, $\exists s.C_1$ and $\exists(u_1, \dots, u_n).P_n$ and no abstract features, it is an $\mathcal{EL}(\mathcal{D})$ concept. The restrictions for $\mathcal{EL}(\mathcal{D})$ concepts ensure polynomial time complexity for standard reasoning tasks. Specifically, as done in [5], we disallow abstract features, since axiom entailment in $\mathcal{EL}$ with functional roles is ExpTime-hard [5].

A *TBox* is a finite set of *general concept inclusion axioms* (GCIs), which are of the form $C \sqsubseteq D$, where $C$ and $D$ are concepts. A *classical ABox* is a finite set of *assertions*, which are of the forms $A(a)$, $s(a, b)$ and $g(a, d)$, where $a, b \in N_i$, $A \in N_c$, $s \in N_r$, $g \in N_{cF}$ and $d \in \Delta^\mathcal{D}$. We call GCIs and assertions collectively *axioms*. A *knowledge base* (KB) $\mathcal{K}$ is a pair $(\mathcal{T}, \mathcal{A})$ of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. Given a KB $\mathcal{K}$, we denote by $\mathrm{sub}(\mathcal{K})$ the *subconcepts* occurring in $\mathcal{K}$. Let $\mathcal{L}$ be a DL, then a TBox/KB that uses only $\mathcal{L}$ concepts is a $\mathcal{L}$ TBox/$\mathcal{L}$ KB.

The semantics of $\mathcal{EL}(\mathcal{D})$ and $\mathcal{ALC}(\mathcal{D})$ is defined in terms of interpretations. An *interpretation* is a tuple $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consisting of a *set of domain elements* $\Delta^\mathcal{I}$ and an *interpretation function* $\cdot^\mathcal{I}$. The *interpretation function* $\cdot^\mathcal{I}$ maps individual names to elements of $\Delta^\mathcal{I}$, concept names to subsets of $\Delta^\mathcal{I}$, concrete features to partial functions $\Delta^\mathcal{I} \to \Delta^\mathcal{D}$, and role names to subsets of $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$ s.t. for all $s \in N_{aF}$, $s^\mathcal{I}$ is a partial function. The extension of $\cdot^\mathcal{I}$ to feature paths is $(s_1 \dots s_n g)^\mathcal{I} = g^\mathcal{I} \circ s_n^\mathcal{I} \circ \dots \circ s_1^\mathcal{I}$, and to (complex) concepts is:

$$\begin{aligned}
\top^\mathcal{I} = \Delta^\mathcal{I} \qquad (\neg C)^\mathcal{I} &= \Delta^\mathcal{I} \setminus C^\mathcal{I} \qquad (C_1 \sqcap C_2)^\mathcal{I} = C_1^\mathcal{I} \cap C_2^\mathcal{I} \\
(\exists s.C)^\mathcal{I} &= \{x \in \Delta^\mathcal{I} \mid \exists y \in \Delta^\mathcal{I} : (x, y) \in s^\mathcal{I} \wedge y \in C^\mathcal{I}\} \\
(\exists(u_1, \dots, u_n).P)^\mathcal{I} &= \{x \in \Delta^\mathcal{I} \mid (u_1^\mathcal{I}(x), \dots, u_n^\mathcal{I}(x)) \text{ is defined and in } P^\mathcal{D}\} \\
(u{\uparrow})^\mathcal{I} &= \{x \in \Delta^\mathcal{I} \mid u^\mathcal{I}(x) \text{ is undefined}\}.
\end{aligned}$$

An axiom $\alpha$ is *true* in an interpretation $\mathcal{I}$, in symbols $\mathcal{I} \models \alpha$, if $\alpha = C \sqsubseteq D$ and $C^\mathcal{I} \subseteq D^\mathcal{I}$, $\alpha = C(a)$ and $a^\mathcal{I} \in C^\mathcal{I}$, $\alpha = s(a, b)$ and $(a^\mathcal{I}, b^\mathcal{I}) \in s^\mathcal{I}$, or $\alpha = g(a, n)$ and $g^\mathcal{I}(a) = n$. An interpretation $\mathcal{I}$ is a *model* of a TBox (an ABox), if all GCIs (assertions) in it are true in $\mathcal{I}$. An interpretation is a model of a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, if it is a model of $\mathcal{T}$ and $\mathcal{A}$. A KB is satisfiable iff it has a model. Given a KB $\mathcal{K}$ and an axiom $\alpha$, we say $\alpha$ *is entailed in* $\mathcal{K}$, in symbols $\mathcal{K} \models \alpha$, iff $\mathcal{I} \models \alpha$ in all models $\mathcal{I}$ of $\mathcal{K}$.

The particular concrete domain to be used needs to be selected carefully, in order to obtain a decidable logic with reasonable complexity bounds. Specifically, axiom entailment with TBoxes already becomes undecidable if $\Delta_{\mathcal{D}} = \mathbb{N}$ and $\Phi_{\mathcal{D}}$ can express incrementation, as well as equality between numbers and with 0[28]. However, by restricting the predicates to basic comparison operators, decidability cannot only be retained, but an increase of complexity for common reasoning tasks can be avoided when adding such concrete domains to the logic. To pursue this as a goal, we concentrate on two concrete domains that allow for standard reasoning in P and ExpTime, respectively. The first concrete domain is $\mathsf{R} = \{\mathbb{R}, \Phi_{\mathsf{R}}\}$ investigated in [25], where $\Phi_{\mathsf{R}}$ contains the binary predicates $\{<, =, >\}$ with the usual semantics, and the unary predicates $\{<_r, =_r, >_r \mid r \in \mathbb{R}\}$, where for $\oplus \in \{<, =, >\}$, the extension is defined as $\oplus_r^{\mathsf{R}} = \{r' \in \mathbb{R} \mid r' \oplus r\}$. This concrete domain allows for axiom entailment in ExpTime, while even small extensions lead to undecidability [25]. The second concrete domain is $\mathsf{R}_> = \{\mathbb{R}, \Phi_{\mathsf{R}_>}\}$, where $\Phi_{\mathsf{R}_>} = \{>_r \mid r \in \mathbb{R}\}$. Since polynomial time reasoning requires the concrete domain to be *convex* [5], we consider this convex concrete domain.

**Example 1.** The axioms in the introduction only use predicates from $\mathsf{R}_>$ and are in the logic $\mathcal{EL}(\mathsf{R}_>)$. Feature paths and the more expressive concrete domain $\mathsf{R}$ allow to compare different values referred to by concrete features. The following more flexible definition of HBP patients compares their diastolic blood pressure (BP) with the maximal diastolic blood pressure assigned to their age group:

$$\exists(\mathsf{diastolicBP}, \mathsf{belongsToAgeGroup\ maxDiastolicBP}).> \sqsubseteq \mathsf{PatientWithHBP}.$$

## 2.1 Queries

We recall atomic, conjunctive and unions of conjunctive queries. Let $N_v$ be a set of variables disjoint from $N_c$, $N_r$, $N_{cF}$ and $N_i$. An *atom* is of the form $C(x)$ or $s(x, y)$, where $C$ is a concept, $s \in N_r$, $x, y \in N_v \cup N_i$. A *conjunctive query (CQ)* $q$ is an expression of the form $\exists x_1, \ldots, x_n : a_1 \wedge \ldots \wedge a_m$, where $x_1, \ldots, x_n \in N_v$ and $a_1, \ldots, a_m$ are atoms. The variables $x_1, \ldots, x_n$ are the *existentially quantified variables in* $q$, the remaining variables in $q$ are the *free variables in* $q$. If a CQ contains only one atom, it is an *atomic query* (AQ). A *union of conjunctive queries (UCQ)* is an expression of the form $q_1 \vee \ldots \vee q_n$, where $q_1, \ldots, q_n$ are CQs with pairwise-disjoint sets of variables. The existentially quantified/free variables of a UCQ are the existentially quantified/free variables of its disjuncts. We call AQs, CQs and UCQs collectively *queries*. A query is *Boolean* if it has no free variables.

Given an interpretation $\mathcal{I}$ and a Boolean CQ $q$, $q$ is *true in* $\mathcal{I}$, in symbols $\mathcal{I} \models q$, iff there is a mapping $\pi$ that maps variables in $q$ to domain elements in $\mathcal{I}$ and each $a \in N_i$ to $a^{\mathcal{I}}$ such that for every atom $A(x)$ in $q$, $\pi(x) \in A^{\mathcal{I}}$, and for every atom $s(x, y)$ in $q$, $(\pi(x), \pi(y)) \in s^{\mathcal{I}}$. A Boolean UCQ is true in $\mathcal{I}$ iff one of its disjuncts is true in $\mathcal{I}$. Finally, given a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and a Boolean query $q$, $q$ *is entailed by* $\mathcal{K}$, in symbols $\mathcal{K} \models q$, if $\mathcal{I} \models q$ in every model of $\mathcal{K}$. The *query entailment problem* is to decide whether a given Boolean query is entailed by a given KB.

The *query answering* problem is to find a substitution from the free variables in the query to individual names such that the resulting Boolean query is entailed by the KB. Because this problem can be polynomially reduced to query entailment, it is typical to focus on the query entailment problem, which is a decision problem, when analysing computational complexity. We follow the same route in this paper.

Note that according to our definition, concrete features cannot be used outside of concepts in a query. Therefore, our queries can only express relations between concrete features that can be captured by a concept in our language. For example, the FOL formula

$$\exists y_1, y_2, z_1, z_2 : s_1(x, y_1) \wedge g_1(y_1, z_1) \wedge s_2(x, y_2) \wedge g_2(y_2, z_2) \wedge z_1 < z_2.$$

| | $\mathcal{EL}(\mathsf{R}_>)$ | | $\mathcal{ALC}(\mathsf{R})$ | |
|---|---|---|---|---|
| | AQs | UCQs | AQs | UCQs |
| Data complexity | P | P | coNP | coNP |
| Combined Complexity | P | NP | ExpTime | ExpTime |

Table 1: Complexity of classical query entailment.

can be captured the query $\exists(s_1 g_1, s_2 g_2).<(x)$, but only given $s_1, s_2 \in N_{aF}$, $g_1, g_2 \in N_{cF}$, and $<$ is a predicate of the concrete domain.

**Example 2.** In a KB with patient records, the following query can be used to retrieve a list of doctors who diagnosed their patients with high blood pressure.

$$\exists y, z : \mathsf{hasPatient}(x, y) \wedge \mathsf{finding}(y, z) \wedge \mathsf{observed}(x, z) \wedge \mathsf{HighBloodPressure}(z)$$

### 2.2 Complexity of Classical Query Entailment

We give tight complexity bounds for query entailment for the introduced DLs. To the best of our knowledge, the complexity of query answering for the logics studied here has not been considered in the literature before. We focus on the DLs $\mathcal{EL}(\mathsf{R}_>)$ and $\mathcal{ALC}(\mathsf{R})$, since $\mathcal{EL}(\mathsf{R})$ has the same expressive power as $\mathcal{ALC}(\mathsf{R})$ [5], and $\mathcal{ALC}(\mathsf{R}_>)$ already has matching lower bounds from $\mathcal{ALC}$ to our upper bounds for $\mathcal{ALC}(\mathsf{R})$. We further assume values from the concrete domain to be represented in binary. Our complexity analysis only concerns knowledge bases that have a finite representation, which by this assumption are those in which each number can be represented with a finite number of bits. When analysing complexity of query entailment, we distinguish between *combined* and *data complexity*, where in combined complexity, the size of the complete input is taken into consideration, while for data complexity, everything but the ABox is fixed.

An overview of the complexities is shown in Table 1. Since the corresponding lower bounds are the same for CQs as for UCQs, we do not include CQs. Matching lower bounds are already known for the DLs $\mathcal{EL}$ and $\mathcal{ALC}$ [35, 33, 10], so that adding the respective concrete domains does not increase the complexity of query answering for these logics. We show in theappendix how to reduce query entailment in $\mathcal{EL}(\mathsf{R}_>)$ to query entailment of $\mathcal{EL}$ KBs, following a technique from [26, Section 2.4]. For $\mathcal{ALC}(\mathsf{R})$, the results are based on and match results from [25],[26, Section 6.2], and [29], which concern the combined complexities of $\mathcal{SHIQ}(\mathsf{R})$ TBox satisfiability and $\mathcal{ALC}(\mathsf{R})$ KB satisfiability, as well as the combined complexity of query entailment in $\mathcal{SHQ}^{\sqcap}$.

## 3 Probabilistic Knowledge Bases with Continuous Probability Distributions

We want to represent both, discrete probabilities of assertions and continuous probability distributions of values of concrete features. As we can simply assign a probability of 1 to assertions that are certain, there is no need to handle certain assertions separately. A *discrete probability assertion* assigns a minimal probability to a classical assertion. This corresponds to the approach taken by *tuple-independent probabilistic database systems* [13], where probabilities are assigned to database and to *ipABoxes* introduced in [22]. For example, the fact that "Otto has a finding that is Diabetes with a probability of at least 0.7" is expressed by the two assertions $\mathsf{finding}(\mathsf{otto}, \mathsf{f1}) : 1$ and $\mathsf{Diabetes}(\mathsf{f1}) : 0.7$.

Note that discrete probability assertions state a lower bound on the probability, rather than the

actual probability, and that statistical independence is only assumed on this lower bound. This way, it is consistent to have the assertions $A(a) : 0.5$, $B(a) : 0.5$ together with the axiom $A \sqsubseteq B$ in the knowledge base. Under our semantics, the probability of $B(a)$ is then higher than 0.5, since this assertion can be entailed due to two different, statistically independent statements in the ABox. Namely, we would infer that the probability of $B(a)$ is at least 0.75 (compare also with [22]).

While for symbolic facts, assigning discrete probabilities is sufficient, for numerical values this is not necessarily the case. For example, if the blood pressure of a patient follows a continuous probability distribution, the probability of it to have any specific value is 0. For this reason, in a *continuous probability assertion*, we connect the value of a concrete feature with a probability density function. This way, the fact that "the diastolic blood pressure of Otto follows a normal distribution with an expected value of 84.5 and a variance of 2" can be expressed by the assertion $\mathsf{diastolicBloodPressure}(\mathsf{otto}) \sim \mathsf{norm}(84.5, 2)$. In addition to a concrete domain $\mathcal{D}$, the DLs introduced in this section are parametrised with a set $\mathcal{P}$ of *probability density functions* (*pdfs*), i.e, Lebesgue-integrable functions $f : A \to \mathbb{R}^+$, with $A \subseteq \mathbb{R}$ being Lebesgue-measurable, such that $\int_A f(x)\,\mathrm{d}x{=}1$ [1].

**Example 3.** As a typical set of probability density functions [1], we define the set $\mathcal{P}_{\mathsf{ex}}$ that contains the following functions, which are parametrised with the numerical constants $\mu, \omega, \lambda, a, b \in \mathbb{Q}$, with $\lambda > 0$ and $a > b$:

**normal distribution** with mean $\mu$ and variance $\omega$:
$\mathsf{norm}(\mu, \omega) : \mathbb{R} \to \mathbb{R}^+,\ x \mapsto \frac{1}{\sqrt{2\pi\omega}} e^{-(x-\mu)^2/2\omega}$,

**exponential distribution** with mean $\lambda$:
$\mathsf{exp}(\lambda) : \mathbb{R}^+ \to \mathbb{R}^+, x \mapsto \lambda e^{-\lambda x}$,

**uniform distribution** between $a$ and $b$:
$\mathsf{uniform}(a, b) : [a, b] \to \mathbb{R}^+, x \mapsto \frac{1}{b-a}$.

Next, we define probabilistic KBs, which consist of a classical TBox and a set of probability assertions.

**Definition 1.** Let $\mathcal{L} \in \{\mathcal{EL}(\mathsf{R}_>), \mathcal{ALC}(\mathsf{R})\}$ and $\mathcal{P}$ be a set of pdfs. A *probabilistic $\mathcal{L}_\mathcal{P}$ ABox* is a finite set of expressions of the form $\alpha : p$ and $g(a) \sim f$, where $\alpha$ is an $\mathcal{L}$ assertion, $p \in [0, 1] \cap \mathbb{D}$,[1] $g \in N_{cF}$, $a \in N_i$, and $f \in \mathcal{P}$. A *probabilistic $\mathcal{L}_\mathcal{P}$ KB* is a tuple $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is an $\mathcal{L}$ TBox and $\mathcal{A}$ is a probabilistic $\mathcal{L}_\mathcal{P}$ ABox. If $\mathcal{P} = \emptyset$, $\mathcal{K}$ and $\mathcal{A}$ are called *discrete*, and if $\mathcal{P} \neq \emptyset$, they are called *continuous*.

## 3.1 Semantics of Probabilistic Knowledge Bases

As typical for probabilistic DLs and databases, we define the semantics using a *possible worlds semantics*. In probabilistic systems that only use discrete probabilities, the possible world semantics can be defined based on finite sets of non-probabilistic data sets, the possible worlds, each of which is assigned a probability [13, 22, 30]. The probability that a query $q$ is entailed then corresponds to the sum of the probabilities of the possible worlds that entail $q$. If continuous probability distributions are used, this approach is insufficient. For example, if the KB contains the assertion $\mathsf{diastolicBP}(p) \sim \mathsf{norm}(84.5, 2)$, the probability of $\mathsf{diastolicBP}(p, x)$ should be 0 for every $x \in \mathbb{R}$. Therefore, we cannot obtain the probability of $\mathsf{diastolicBP}(p) > 85$ by just adding the probabilities of the possible worlds that entail $\mathsf{diastolicBP}(p, x)$ for some $x > 85$. To

[1] Here, the set $\mathbb{D} \subseteq \mathbb{R}$ denotes the *dyadic rationals*, that is, the set of all real numbers that have a finite number of bits after the binary point.

overcome this problem, we assign probabilities to (possibly uncountable) *sets* of possible worlds, rather than to single possible worlds. Specifically, we define the semantics using continuous probability measure spaces [1]. A *measure space* is a tuple $M = (\Omega, \Sigma, \mu)$ with $\Sigma \subseteq 2^\Omega$ and $\mu : \Sigma \to \mathbb{R}$ such that

1. $\Omega \in \Sigma$ and $\Sigma$ is closed under complementation, countable unions and countable intersections,
2. $\mu(\emptyset) = 0$, and
3. $\mu(\bigcup_{E \in \Sigma'}) = \sum_{E \in \Sigma'} \mu(f)$ for every countable set $\Sigma' \subseteq \Sigma$ of pair-wise disjoint sets.

If additionally $\mu(\Omega) = 1$, $M$ is a *probability measure space*.

We define a probability measure space $M_\mathcal{A} = (\Omega_\mathcal{A}, \Sigma_\mathcal{A}, \mu_\mathcal{A})$ that captures the relevant probabilities in a probabilistic ABox $\mathcal{A}$, similar to how it is done in [22] for discrete probabilistic ABoxes. For this, we introduce the three components $\Omega_\mathcal{A}$, $\Sigma_\mathcal{A}$ and $\mu_\mathcal{A}$ one after another. For simplicity, we assume all pdfs $f : A \to \mathbb{R} \in \mathcal{P}$ to be extended to the full real line by setting $f(x) = 0$ for all $x \in \mathbb{R} \setminus A$.

Given a probabilistic ABox $\mathcal{A}$, the set of *possible worlds for* $\mathcal{A}$, in symbols $\Omega_\mathcal{A}$, consists of all classical ABoxes $w$ such that for every $g(a) \sim f \in \mathcal{A}$, $w$ contains $g(a, x)$ for some $x \in \mathbb{R}$, and for every axiom $\alpha \in w$, either $\alpha : p \in \mathcal{A}$, or $\alpha$ is of the form $g(a, x)$ and $g(a) \sim f \in \mathcal{A}$. For $w \in \Omega_\mathcal{A}$, we write $w \models g(a) \oplus x$, $x \in \mathbb{R}$, $\oplus \in \{<, \leq, =, \geq, >\}$, iff $w \models g(a, y)$ and $y \oplus x$. We write $w \models g(a) \oplus h(b)$ iff $w \models g(a, y), h(b, z)$ and $y \oplus z$. We abbreviate $w \models g(a) \geq x, g(a) \leq y$ by $w \models g(a) \in [x, y]$. The *event space over* $\Omega_\mathcal{A}$, in symbols $\Sigma_\mathcal{A}$, is now the smallest subset $\Sigma_\mathcal{A} \subseteq 2^{\Omega_\mathcal{A}}$ that satisfies the following conditions:

1. $\Omega_\mathcal{A} \in \Sigma_\mathcal{A}$,
2. for every $\alpha : p \in \mathcal{A}$, $\{w \in \Omega_\mathcal{A} \mid \alpha \in w\} \in \Sigma_\mathcal{A}$,
3. for every $g(a) \sim f \in \mathcal{A}$, $x \in \mathbb{R}$, $\{w \in \Omega_\mathcal{A} \mid w \models g(a) < x\} \in \Sigma_\mathcal{A}$,
4. for every $g_1(a_1) \sim f_1$, $g_2(b) \sim f_2 \in \mathcal{A}$, $\{w \in \Omega_\mathcal{A} \mid w \models g_1(a) < g_2(b)\} \in \Sigma_\mathcal{A}$, and
5. $\Sigma_\mathcal{A}$ is closed under complementation, countable unions and countable intersections.

The conditions ensure that for every query $q$ and TBox $\mathcal{T}$, the set of possible worlds $w$ such that $(\mathcal{T}, w) \models q$ is included in $\Sigma_\mathcal{A}$. To complete the definition of the measure space, we now assign probabilities to these sets via the measure function $\mu_\mathcal{A}$. This function has to respect the probabilities expressed by the discrete and continuous probability assertions in $\mathcal{A}$, as well as the assumption that these probabilities are statistically independent. We define $\mu_\mathcal{A}$ explicitly for sets of possible worlds that are selected by the assertions in them, and by upper bounds on the concrete features occurring in continuous probability assertions. By additionally requiring that Condition 3 in the definition of measure spaces is satisfied for $\mu_\mathcal{A}$, this is sufficient to fix the probability for any set in $\Sigma_\mathcal{A}$.

Given a probabilistic ABox $\mathcal{A}$, we denote by $\mathsf{cl\text{-}ass}(\mathcal{A}) = \{\alpha \mid \alpha : p \in \mathcal{A}\}$ the classical assertions occurring in $\mathcal{A}$. A *bound set for* $\mathcal{A}$ is a set $\mathbf{B}$ of inequations of the form $g(a) < x$, $x \in \mathbb{R}$, where $g(a) \sim f \in \mathcal{A}$ and every concrete feature $g(a)$ occurs at most once in $\mathbf{B}$. Given a set $\mathcal{E} \subseteq \mathsf{cl\text{-}ass}(\mathcal{A})$ of assertions from $\mathcal{A}$ and a bound set $\mathbf{B}$ for $\mathcal{A}$, we define the corresponding set $\Omega_\mathcal{A}^{\mathcal{E},\mathbf{B}}$ of possible worlds in $\Omega_\mathcal{A}$ as

$$\Omega_\mathcal{A}^{\mathcal{E},\mathbf{B}} = \{w \in \Omega_\mathcal{A} \mid w \cap \mathsf{cl\text{-}ass}(\mathcal{A}) = \mathcal{E}, w \models \mathbf{B}\}.$$

The probability measure space for $\mathcal{A}$ is now the probability measure space $M_{\mathcal{A}} = (\Omega_{\mathcal{A}}, \Sigma_{\mathcal{A}}, \mu_{\mathcal{A}})$, such that for every $\mathcal{E} \subseteq \mathsf{cl\text{-}ass}(\mathcal{A})$ and every bound set $\mathbf{B}$ for $\mathcal{A}$,

$$\mu_{\mathcal{A}}(\Omega_{\mathcal{A}}^{\mathcal{E},\mathbf{B}}) = \prod_{\substack{\alpha:p\in\mathcal{A}\\ \alpha\in\mathcal{E}}} p \cdot \prod_{\substack{\alpha:p\in\mathcal{A}\\ \alpha\notin\mathcal{E}}} (1-p) \cdot \prod_{\substack{g(a)\sim f\in\mathcal{A}\\ g(a)<x\in\mathbf{B}}} \int_{-\infty}^{x} f(y)\,\mathrm{d}y.$$

As shown in the appendix, this definition uniquely determines $\mu_{\mathcal{A}}(W)$ for all $W \in \Sigma_{\mathcal{A}}$, including for sets such as $W = \{w \in \Omega_{\mathcal{A}} \mid w \models g_1(a) < g_2(b)\}$. The above product is a generalisation of the corresponding definition in [22] for discrete probabilistic KBs, where in addition to discrete probabilities, we take into consideration the continuous probability distribution of the concrete features in $\mathcal{A}$. Recall that if a concrete feature $g(a)$ follows the pdf $f$, the integral $\int_{-\infty}^{x} f(y)\,\mathrm{d}y$ gives us the probability that $g(a) < x$.

Since we have now finished the formal definition of the semantics of probabilistic ABoxes, we can now define the central reasoning task studied in this paper. As in Section 2.1, we concentrate on probabilistic query entailment rather than on probabilistic query answering. The latter is a ranked search problem that can be polynomially reduced to probabilistic query entailment as in [22]. Based on the measure space $M_{\mathcal{A}}$, we define the *probability of a Boolean query* $q$ in a probabilistic KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ as $P_{\mathcal{K}}(q) = \mu_{\mathcal{A}}(\{w \in \Omega_{\mathcal{A}} \mid (\mathcal{T}, w) \models q\})$. Note that due to the open-world assumption, strictly speaking, $P_{\mathcal{K}}(q)$ corresponds to a lower bound on the probability of $q$, since additional facts may increase the value of $P_{\mathcal{K}}(q)$.

Different to [22] and classical approaches in probabilistic query answering, because $\mathcal{P}$ contains real functions, $P_{\mathcal{K}}(q)$ is in general a real number, and as such not finitely representable. In practice, it is typical and usually sufficient to compute approximations of real numbers. To capture this adequately, we take the required precision of the probability $P_{\mathcal{K}}(q)$ as additional input to the probabilistic query entailment problem. For a real number $x \in \mathbb{R}$ and $n \in \mathbb{N}$, we use the notation $\langle x \rangle_n$ to refer to an *$n$-bit approximation of $x$*, that is, a real number such that $|\langle x \rangle_n - x| < 2^{-n}$. Note that, while we do not enforce it, generally $n$ bits after the binary point are sufficient to identify $\langle x \rangle_n$. We can now state the main reasoning problem studied in this paper.

**Definition 2.** The *probabilistic query entailment problem* is the problem of computing, given a probabilistic KB $\mathcal{K}$, a Boolean query $q$ and a natural number $n$ in unary encoding, a number $x$ s.t. $x = \langle P_{\mathcal{K}}(q) \rangle_n$.

Since the precision parameter $n$ determines the size of the result, we assume it in unary encoding. If we would represent it in binary, it would already take exponential time just to write the result down.

# 4 Feasibility Conditions for PDFs

Up to now, we did not put any restrictions on the set $\mathcal{P}$ of pdfs, so that a given set $\mathcal{P}$ could easily render probabilistic query entailment uncomputable. In this section, we define a set of feasibility conditions on pdfs that ensure that probabilistic query entailment is not computationally harder than when no continuous probability distributions are used. We know from results in probabilistic databases [13], that query entailment over probabilistic data is #·P-hard. Note that integration of pdfs over bounded intervals can be reduced to probabilistic query answering. Namely, if $g(a) \sim f \in \mathcal{A}$, we have $P_{(\emptyset,\mathcal{A})}((\exists g.{>}_r)(a)) = \int_r^{\infty} f(x)\,\mathrm{d}y$ for all $r \in \mathbb{R}$. Our feasibility conditions ensure that the complexity of approximating integrals does not dominate the overall complexity of probabilistic query entailment.

We first recall some notions from the complexity theory of real functions by Ker-I Ko [23], which identifies computability of real numbers $x \in \mathbb{R}$ and functions $f : A \to \mathbb{R}$, $A \subseteq \mathbb{R}$, with the computability of $n$-bit approximations $\langle x \rangle_n$ and $\langle f(x) \rangle_n$, where $n$ is given in unary encoding. Since real function arguments have no finite representation in general, computable real functions are modelled as function oracle Turing machines $T^{\phi(x)}$, where the oracle $\phi(x)$ represents the function argument $x$ and can be queried for $n$-bit approximations $\langle x \rangle_n$ in time linear in $c + n$, where $c$ is the number of bits in $x$ before the binary point. Given a precision $n$ in unary encoding on the input tape, $T^{\phi(x)}$ then writes a number $\langle f(x) \rangle_n$ on the output tape. This formalism leads to a natural definition of computability and complexity of real numbers and real functions. Namely, a real number $x \in \mathbb{R}$ is P-*computable* iff there is a polynomial time Turing machine that computes a function $\phi : \mathbb{N} \mapsto \mathbb{D}$ s.t. $\phi(n) = \langle x \rangle_n$. A function $f : A \to \mathbb{R}$, $A \subseteq \mathbb{R}$, is P-computable iff there is a function oracle Turing machine $T^{\phi(x)}$ as above that computes for all $x \in A$ a function $\psi : \mathbb{N} \mapsto \mathbb{D}$ with $\psi(n) = \langle f(x) \rangle_n$ in time polynomial in $n$ and the number of bits in $x$ before the binary point.

An important property of P-computable functions $f$ that we use in the next section is that they have a monotone and polynomial *modulus of continuity* (*modulus*), that is, a monotone, polynomial function $\omega_f : \mathbb{N} \to \mathbb{N}$ s.t. for all $n \in \mathbb{N}$ and $x, y \in [2^{-n}, 2^n]$, $|x - y| < 2^{-\omega_f(n)}$ implies $|f(x) - f(y)| < 2^{-n}$ [21, 23, Chapter 3].

Approximating integrals $\int_0^1 f(x)\,\mathrm{d}x$ of P-computable functions $f : [0, 1] \to \mathbb{R}$ is $\#\cdot$P-complete [23, Chapter 5]. To be able to integrate over unbounded integrals in $\#\cdot$P, we introduce an additional condition.

**Definition 3.** A probability density function $f$ is $\#\cdot$P-*admissible* iff it satisfies the following conditions:

1. $f$ is P-computable, and
2. there is a monotone polynomial function $\delta_f : \mathbb{N} \to \mathbb{N}$ such that for all $n \in \mathbb{N}$:

$$1 - \int_{-2^{\delta_f(n)}}^{2^{\delta_f(n)}} f(x)\,\mathrm{d}x < 2^{-n}.$$

Condition 2 allows us to reduce integration over *un*bounded integrals to integration over bounded integrals: to obtain a precision of $n$ bits, it is sufficient to integrate inside the interval $[-2^{\delta_f(n)}, 2^{\delta_f(n)}]$. Note that as a consequence of Condition 1, there is also a polynomial $\rho_f : \mathbb{N} \to \mathbb{N}$ s.t. for all $x \in [-2^{\delta_f(n)}, 2^{\delta_f(n)}]$, $f(x) < 2^{\rho_f(n)}$. Otherwise, approximations of $f(x)$ would require a number of bits that is not polynomially bounded by the number of bits in $x$ before the binary point, and could thus not be computed in polynomial time. We call $\delta_f$ and $\rho_f$ respectively *bounding function* and *range function* of $f$. In the following, we assume that for any set $\mathcal{P}$ of $\#\cdot$P-admissible pdfs, their moduli, bounding functions and range functions are known.

The above properties are general enough to be satisfied by most common pdfs. Specifically, we have the following lemma for the set $\mathcal{P}_{\mathsf{ex}}$ defined in Example 3:

**Lemma 1.** *Every function in* $\mathcal{P}_{\mathsf{ex}}$ *is* $\#\cdot$P-*admissible.*

# 5 Complexity of Probabilistic Query Answering

We study the complexity of probabilistic query answering for KBs with $\#\cdot$P-admissible pdfs. As often in probabilistic reasoning, counting complexity classes play a central role in our study.

However, strictly speaking, these are defined for computation problems for *natural numbers*. To get a characterisation for probabilistic query answering, we consider corresponding counting problems. Their solutions are obtained by, intuitively, shifting the binary point of an approximated query probability to the right to obtain a natural number. We first recall counting complexity classes following [19].

**Definition 4.** Let $\mathcal{C}$ be a class of decision problems. Then, $\#\cdot\mathcal{C}$ describes the class of functions $f : A \to \mathbb{N}$ such that

$$f(x) = \big\|\{y \mid R(x,y) \wedge |y| < p(|x|)\}\big\|$$

for some $\mathcal{C}$-decidable relation $R$ and polynomial function $p$.

Relevant to this section are the counting complexity classes $\#\cdot$P, $\#\cdot$NP and $\#\cdot$coNP. The class $\#\cdot$P is also called #P. The following inclusions are known: $\#\cdot P \subseteq \#\cdot\text{NP} \subseteq \#\cdot\text{coNP} \subseteq \text{FPSpace}$ [19].

In order to characterise the complexity of probabilistic query answering using counting classes, we consider corresponding counting problems, inspired by [23, Chapter 5] and [13]. For a function $f : A \to \mathbb{D}$, we call $g : A \to \mathbb{N}$ a *corresponding counting problem* if $g(x) = 2^{p(x)} f(x)$ for all $x \in A$, where $p : A \to \mathbb{N}$ and $p$ can be computed in unary in polynomial time.[2]

For discrete probabilistic KBs, the above definition allows us to give a complexity upper bound for a counting problem corresponding to probabilistic query entailment in a quite direct way. Without loss of generality, we assume that queries contain only concept names as concepts. If $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is discrete, the probability measure space $M_{\mathcal{A}}$ has only a finite set $\Omega_{\mathcal{A}}$ of possible worlds, and each possible world $w \in \Omega_{\mathcal{A}}$ has a probability $\mu_{\mathcal{A}}(\{w\})$ that can be represented with a number of bits polynomial in the size of the input. We use this to define a relation $R$ as used in Definition 4. Let $b_{\mathcal{K}}$ be the maximal number of bits used by any probability $\mu_{\mathcal{A}}(\{w\})$, $w \in \Omega_{\mathcal{A}}$. Define the relation $R$ by setting $R((\mathcal{K}, q, n), (w, d))$ for all $w \in \Omega_{\mathcal{A}}$, $d \in \mathbb{N}$ s.t. $(\mathcal{T}, w) \models q$ and $d < 2^{b_{\mathcal{K}}} \cdot \mu_{\mathcal{A}}(\{w\})$, where $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. One easily establishes that $\langle P_{\mathcal{K}}(q)\rangle_n = 2^{-b_{\mathcal{K}}} \cdot \|\{y \mid R((\mathcal{K}, q, n), y)\}\|$ for any $n \in \mathbb{N}$. (Note that our "approximation" is always the precise answer in this case.) For discrete KBs, we thus obtain a complexity upper bound of $\#\cdot\mathcal{C}$ for the corresponding counting problem defined by $g(\mathcal{K}, q, n) = 2^{b_{\mathcal{K}}} \cdot P_{\mathcal{K}}(q)$, where $\mathcal{C}$ is the complexity of classical query entailment.

In order to transfer this approach to continuous probabilistic KBs, we define a discretisation of continuous probability measure spaces based on the precision parameter $n$ and the TBox $\mathcal{T}$. Namely, given a probabilistic KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and a desired precision $n$, we step-wise modify the measure space $M_{\mathcal{A}}$ into an approximated measure space $M^a_{\mathcal{K},n} = (\Omega^a_{\mathcal{K},n}, \Sigma^a_{\mathcal{K},n}, \mu^a_{\mathcal{K},n})$ such that (i) the size of each possible world $w \in \Omega^a_{\mathcal{K},n}$ is polynomially bounded by $|\mathcal{K}| + n$, (ii) for each $w \in \Sigma^a_{\mathcal{K},n}$, $\mu^a_{\mathcal{K},n}(\{w\})$ can be computed precisely and in time polynomial in $|\mathcal{K}| + n$, and (iii) it holds $\mu^a_{\mathcal{K},n}(\{w \in \Omega^a_{\mathcal{K},n} \mid (\mathcal{T}, w) \models q\}) = \langle P_{\mathcal{K}}(q)\rangle_n$ for every query $q$. Real numbers occur in $M_{\mathcal{A}}$ in concrete feature values and in the range of $\mu_{\mathcal{A}}$, and have to be replaced by numbers with a polynomially bounded number of bits. We proceed in three steps: (1) we first reduce the number of bits that occur *before* the binary point in any concrete feature value, (2) we then reduce the number of bits that occur *after* the binary point in any concrete feature value, and (3) we finally reduce the number of bits in the range of $\mu_{\mathcal{A}}$.

We define $\mathbf{C} = \{g_i(a_i) \sim f_i \in \mathcal{A}\}$ as the set of continuous probability assertions in $\mathcal{K}$ and $\mathcal{F} = \{f_i \mid g_i(a_i) \sim f_i \in \mathbf{C}\}$ as the relevant pdfs in $\mathcal{K}$. We also set $n_v = \|\mathbf{C}\|$ and $n_c$ as the number of unary concrete domain predicates in $\mathcal{K}$.

**Step 1: Reduce the number of bits before the binary point.** Because every function

[2] Note that the counting complexity classes considered here are all closed under this operation. To see this, consider $f$ and $g$ characterized by the relations $R$ and $R'$ s.t. $R' = \{(x, y\#z) \mid R(x,y), z \in \{0,1\}^*, |z| = p(x)\}$. Clearly, $g(x) = 2^{p(x)} f(x)$.

$f \in \mathcal{F}$ has a monotone polynomial bounding function, we can obtain a function $\delta : \mathbb{N} \to \mathbb{N}$ s.t. for every pdf $f \in \mathcal{F}$ and every $n' \in \mathbb{N}$, we have

$$1 - \int_{-2^{\delta(n')}}^{2^{\delta(n')}} f(x)\,\mathrm{d}x < 2^{-n'}.$$

The first step is to remove all possible worlds $w$ in which for some $g(a) \sim f \in \mathbf{C}$, we have $w \not\models g(a) \in [-2^{\delta(n_v+n)}, 2^{\delta(n_v+n)}]$. Note that for each $g(a) \sim f \in \mathcal{A}$, the probability of $g(a)$ to lay outside this interval is $2^{-n_v-n}$. Based on this, one can show that for the resulting measure space $M_1 = (\Omega_1, \Sigma_1, \mu_1)$, we have $|\mu_{\mathcal{A}}(\Omega_{\mathcal{A}}) - \mu_1(\Omega_1)| < 2^{-n-1}$. This restricts also the overall error on the probability of any query. Therefore, we have a remaining error of $2^{-n-1}$ that we can make in subsequent steps. Note that the number of bits before the binary point in any concrete feature value is now polynomially bounded by the input.

**Step 2: Reduce the number of bits after the binary point.** Intuitively, in this step we "replace" each possible world $w \in \Omega_1$ by a possible world $w'$ that is obtained by "cutting off" in all concrete feature values all digits after a certain position after the binary point, preserving its probability. First, we specify the maximum number $m$ of digits after the binary point we keep. Similar as for the bounding function $\delta$, we can obtain a polynomial function $\omega$ that is a modulus of all functions $f \in \mathcal{F}$, and a polynomial function $\rho$ that is a range function of all functions $f \in \mathcal{F}$. Let $k = \rho(n_v + n)$ be the highest number of bits before the binary point in the range of any pdf in the remaining interval $[-2^{\delta(n+n_v)}, 2^{\delta(n+n_v)}]$, and set $l = n_v + \delta(n_v + n) + 2 + n$. Based on $k$, $l$ and $\omega$, we define the maximal precision $m$ by

$$m = \lceil \log_2(n_v(n_v + n_c)) + k + n + 3 + \omega(l) \rceil .$$

The motivation behind this definition will become clear in the following. For now, just notice that $m$ is polynomially bounded by $|\mathcal{K}| + n$.

In the approximated measure space $M_2 = (\Omega_2, \Sigma_2, \mu_2)$, $\Omega_2$ contains all worlds from $\Omega_1$ in which each concrete feature value has at most $m$ bits after the binary point. To preserve the probabilities, we define a function $\Omega_{2\to1} : \Omega_2 \to 2^{\Omega_1}$ that maps each possible world $w \in \Omega_2$ to the possible worlds in $\Omega_1$ that have been "replaced" by $w$. $\Omega_{1\to2}$ is defined as

$$\begin{aligned}\Omega_{2\to1}(w) = \{w' \in \Omega_1 \mid & w \cap \mathsf{cl\text{-}ass}(\mathcal{A}) = w' \cap \mathsf{cl\text{-}ass}(\mathcal{A}),\\ & \forall g(a,x) \in w, g(a) \sim f \in \mathbf{C} : w' \models g(a) \in [x, x + 2^{-m}]\}.\end{aligned}$$

The measure function $\mu_2$ is now defined by

$$\mu_2(\{w\}) = \mu_1(\Omega_{2\to1}(w)).$$

This transformation affects the probability of concepts such as $\exists(g_1, g_2).{>}$ and $\exists g.{>}_r$, because the probability that two concrete features have the same value, or that a concrete feature has a value occurring in some unary domain predicate, increases. One can show that this probability is bounded by $n_v(n_v + n_c) \cdot 2^{-m+k+1}$. By definition, $m > \log_2(n_v(n_v + n_c)) + k + n + 3$, so that the error created in this step is bounded by $2^{-n-2}$.

**Step 3: Reduce the number of bits in the probabilities.** Each possible world $M_2$ can be finitely represented and has a size that is polynomially bounded in the size of the input. However, the probabilities for each possible world are still real numbers. We first explain how we approximate the probabilities for a single concrete feature. For an assertion $g_i(a_i) \sim f_i \in \mathbf{C}$, and a number $x \in \mathbb{R}$ with $m$ bits after the binary point, we have $\mu_2(\{w \in \Omega_2 \mid w \models g(a) = x\}) = \int_x^{x+2^{-m}} f_i(y)\,\mathrm{d}y$. To discretise this probability, we make use of the modulus $\omega$ of the pdfs used in $\mathcal{K}$. Recall that, by the definition of a modulus, for any precision $n' \in \mathbb{N}$ and two real numbers $x, y \in [2^{-n'}, 2^{n'}]$, $|x - y| < 2^{-\omega(n')}$ implies $|f_i(x) - f_i(y)| < 2^{-n'}$. By construction, we

| | $\mathcal{EL}(\mathsf{R}_>)_\mathcal{P}$ | | $\mathcal{ALC}(\mathsf{R})_\mathcal{P}$ | |
|---|---|---|---|---|
| | AQs | UCQs | AQs | UCQs |
| Data complexity | $\#\cdot$P | $\#\cdot$P | $\#\cdot$coNP | $\#\cdot$coNP |
| Combined Complexity | $\#\cdot$P | $\#\cdot$NP | ExpTime | ExpTime |

Table 2: Complexities of counting problems corresponding to prob. query entailment.

have $m > \omega(l)$, and hence, for $x \in [2^{-l}, 2^l]$ and $y \in [x, x + 2^{-m}]$, we have $|f_i(x) - f_i(y)| < 2^{-l}$. Consequently, the integral $\int_x^{x+2^{-m}} f_i(y)\,\mathrm{d}y$ can be approximated by the product $2^{-m} \cdot \langle f_i(x)\rangle_l$, and we have

$$\left| \int_x^{x+2^{-m}} f_i(y)\,\mathrm{d}y - 2^{-m} \cdot \langle f_i(x)\rangle_l \right| \;<\; 2^{-m-l}.$$

There are $2^{\delta(n_v+n)+1+m}$ different values per concrete feature in our measure space, so that an error of $2^{-m-l}$ per approximated interval introduces a maximal error of $2^{-n-n_v-1}$ for each concrete feature value (recall $l = n_v + \delta(n_v + n) + 2 + n$). If we approximate all pdfs this way, for similar reasons as in Step 1, we obtain a maximal additional error of $2^{-n-2}$ for any query.

Based on these observations, we define the final discretised measure space. Specifically, we define the measure space $M^a_{\mathcal{K},n} = (\Omega^a_{\mathcal{K},n}, \Sigma^a_{\mathcal{K},n}, \mu^a_{\mathcal{K},n})$, where $\Omega^a_{\mathcal{K},n} = \Omega_2$ and $\mu^a_{\mathcal{K},n}$ is specified by

$$\mu^a_{\mathcal{K},n}(\{w\}) \;=\; \prod_{\substack{\alpha:p\in\mathcal{A}\\ \alpha\in w}} p \cdot \prod_{\substack{\alpha:p\in\mathcal{A}\\ \alpha\notin w}} (1-p) \cdot \prod_{\substack{g(a)\sim f\in\mathcal{A}\\ g(a,x)\in w}} 2^{-m}\langle f(x)\rangle_l.$$

Note that $\mu^a_{\mathcal{K},n}(\{w\})$ can be evaluated in polynomial time, and can be represented with at most $2 + n_a \cdot n_b + n_v \cdot (m + l)$ bits, where $n_a$ is the number of discrete probability assertions and $n_b$ the maximal number of bits in a discrete probability assertion.

Given a probabilistic KB $\mathcal{K}$ and a precision $n \in \mathbb{N}$, we call the measure space $M^a_{\mathcal{K},n}$ constructed above the *$n$-approximated probability measure space* for $\mathcal{K}$. We have the following lemma.

**Lemma 2.** *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a probabilistic KB, $q$ a query, $n \in \mathbb{N}$ and $M^a_{\mathcal{K},n}$ the $n$-approximated probability measure space for $\mathcal{K}$. Then,*

$$\mu^a_{\mathcal{K},n}(\{w \in \Omega^a_{\mathcal{K},n} \mid (\mathcal{T}, w) \models q\}) = \langle P_\mathcal{K}(q)\rangle_n.$$

Note that one can test in polynomial time whether a given possible world is in $\Omega^a_{\mathcal{K},n}$, and compute its probability in polynomial time. Using the observations from the beginning of this section, together with the complexity results in Table 1, we can establish the upper bounds for data and combined complexity shown in Table 2 on counting problems corresponding to probabilistic query answering, which already hold for discrete probabilistic KBs without concrete domain. To the best of our knowledge, only the data complexity for query answering in probabilistic $\mathcal{EL}$ has been considered in the literature before [22], while the other results are new. For the ExpTime upper bounds, note that the approximated measure space has at most exponentially many elements, and can thus be constructed and checked in exponential time.

Hardness for all complexities already holds for discrete probabilistic KBs, so that continuous, $\#\cdot$P-admissible probability distributions do not increase the complexity of probabilistic query answering. A general $\#\cdot$P-lower bound follows from the corresponding complexity of probabilistic query entailment in probabilistic databases [13], while for the combined complexities in $\mathcal{ALC}(\mathsf{R})_\mathcal{P}$, the lower bound follows from the non-probabilistic case. For the remaining complexities, we provide matching lower bounds for the corresponding counting problems in the appendix using appropriate reductions. Specifically, we show $\#\cdot$NP-hardness w.r.t. combined complexity under *subtractive reductions* in the case of UCQ entailment in $\mathcal{EL}$, and $\#\cdot$coNP-hardness w.r.t data complexity under *parsimonious reductions* in the case of AQ entailment in $\mathcal{ALC}$ [16].

# 6 Conclusion

When numerical data are of an uncertain nature, such as data obtained by sensor readings or video tracking, they can often be more precisely represented using continuous probability distributions than using discrete distributions. While there is work on OBQA for discrete probabilistic KBs in DL-Lite and $\mathcal{EL}$ [22], this is the first work that considers KBs with concrete domains and continuous probability distributions. For our complexity analysis, we devised a set of feasibility conditions for probability distributions based on the complexity theory of real functions, which captures most typical distributions one might encounter in realistic applications. We show that under these conditions, continuous probability distributions do not increase the complexity of probabilistic query entailment. Using a similar technique as in [23, Chapter 5], our results can likely be extended to a wider class of probability distributions, where the requirement of P-computability is weakened to *polynomial approximability*.

For light-weight description logics, it is often possible to rewrite queries w.r.t the ontology, so that they can be answered directly by a corresponding database system. As there are probabilistic database systems like Orion 2.0 that support continuous probability distributions [36], query rewriting techniques for continuous probabilistic KBs could be employed in our setting as well. For more expressive DLs, a practical implementation could be based on a less fine-grained representation of measure spaces, for which relevant intervals for each concrete feature value are determined based on the concrete domain predicates in the TBox. Probabilities could then be computed using standard algorithms for numerical integration. It might also be worth investigating whether Monte-Carlo approximations can be used for practical implementations. However, as observed in [22], this might be hard to accomplish already for discrete probabilistic $\mathcal{EL}$ KBs. Another basis for practical implementations could be approximation techniques developed for other logical frameworks involving continuous probability distributions, such as the one presented in [7].

# References

[1] Malcolm Ritchie Adams and Victor Guillemin. *Measure theory and probability*. Springer, 1996.

[2] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The *DL-Lite* family and relations. *J. Artif. Intell. Res.*, 36:1–69, 2009.

[3] Alessandro Artale, Vladislav Ryzhikov, and Roman Kontchakov. *DL-Lite* with attributes and datatypes. In *Proc. ECAI'12*, pages 61–66. IOS Press, 2012.

[4] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. Query rewriting for *DL-Lite* with $n$-ary concrete domains, 2017. to appear in Proc. IJCAI'17.

[5] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. IJCAI'05*, pages 364–369. Professional Book Center, 2005.

[6] Franz Baader and Philipp Hanschke. A scheme for integrating concrete domains into concept languages. In *Proc. IJCAI'91*, pages 452–457, 1991.

[7] Vaishak Belle, Guy Van den Broeck, and Andrea Passerini. Hashing-based approximate probabilistic inference in hybrid domains: An abridged report. In *Proc. IJCAI'16*, pages 4115–4119, 2016.

[8] Richard P. Brent. The complexity of multiple-precision arithmetic. In *The Complexity of Computational Problem Solving*, pages 126–165. University of Queensland Press, 1976.

[9] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reas.*, 39(3):385–429, 2007.

[10] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data complexity of query answering in description logics. *Artificial Intelligence*, 195:335 – 360, 2013.

[11] İsmail İlkan Ceylan, Thomas Lukasiewicz, and Rafael Peñaloza. Complexity results for probabilistic datalog±. In *Proc. ECAI'16*, pages 1414–1422. IOS Press, 2016.

[12] İsmail İlkan Ceylan and Rafael Peñaloza. Probabilistic query answering in the Bayesian description logic $\mathcal{BEL}$. In *Proc. SUM'15*, pages 21–35. Springer, 2015.

[13] Nilesh Dalvi and Dan Suciu. Management of probabilistic data: foundations and challenges. In *Proc. SIGMOD'07*, pages 1–12. ACM, 2007.

[14] Claudia D'Amato, Nicola Fanizzi, and Thomas Lukasiewicz. Tractable reasoning with Bayesian description logics. In *Proc. SUM'08*, pages 146–159. Springer, 2008.

[15] Waltenegus Dargie. The role of probabilistic schemes in multisensor context-awareness. In *Proc. PerCom'07*, pages 27–32. IEEE, 2007.

[16] Arnaud Durand, Miki Hermann, and Phokion G Kolaitis. Subtractive reductions and complete problems for counting complexity classes. *Theoretical Computer Science*, 340(3):496–513, 2005.

[17] Peter L Elkin, Steven H Brown, Casey S Husser, Brent A Bauer, Dietlind Wahner-Roedler, S Trent Rosenbloom, and Ted Speroff. Evaluation of the content coverage of SNOMED CT: ability of SNOMED clinical terms to represent clinical problem lists. *Mayo Clin. Proc.*, 81(6):741–748, 2006.

[18] Birte Glimm, Carsten Lutz, Ian Horrocks, and Ulrike Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *J. Artif. Intell. Res. (JAIR)*, 31:157–204, 2008.

[19] Lane A Hemaspaandra and Heribert Vollmer. The satanic notations: counting classes beyond #P and other definitional adventures. *ACM SIGACT News*, 26(1):2–13, 1995.

[20] André Hernich, Julio Lemos, and Frank Wolter. Query answering in DL-Lite with datatypes: A non-uniform approach. In *Proc. AAAI'17*, 2017.

[21] H James Hoover. Feasible real functions and arithmetic circuits. *SIAM Journal on Computing*, 19(1):182–204, 1990.

[22] Jean Christoph Jung and Carsten Lutz. Ontology-based access to probabilistic data with OWL QL. In *Proc. ISWC'12*, pages 182–197. Springer, 2012.

[23] Ker-I Ko. *Complexity Theory of Real Functions*. Birkhäuser, 1991.

[24] Nilay Kumar, Monica Khunger, Arjun Gupta, and Neetika Garg. A content analysis of smartphone–based applications for hypertension management. *Journal of the American Society of Hypertension*, 9(2):130–136, 2015.

[25] Carsten Lutz. Adding numbers to the $\mathcal{SHIQ}$ description logic—first results. In *Proc. KR'01*, pages 191–202. Citeseer, 2001.

[26] Carsten Lutz. *The complexity of description logics with concrete domains*. PhD thesis, RWTH Aachen, 2002.

[27] Carsten Lutz. Description logics with concrete domains—a survey. In *Advances in Modal Logic 4*, pages 265–296. King's College Publications, 2002.

[28] Carsten Lutz. NExpTime-complete description logics with concrete domains. *ACM Transactions on Computational Logic (TOCL)*, 5(4):669–705, 2004.

[29] Carsten Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proc. IJCAR'08*, pages 179–193. Springer, 2008.

[30] Carsten Lutz and Lutz Schröder. Probabilistic description logics for subjective uncertainty. In *Proc. KR'10*, pages 393–403. AAAI Press, 2010.

[31] Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *Proc. IJCAI'09*, pages 2070–2075. IJCAI/AAAI, 2009.

[32] A.L. Rector, A. Gangemi, E. Galeazzi, A.J. Glowinski, and A. Rossi-Mori. The GALEN CORE model schemata for anatomy: Towards a re-usable application-independent model of medical concepts. In *Proc. MIE'94*, pages 229–233, 1994.

[33] Riccardo Rosati. On conjunctive query answering in $\mathcal{EL}$. In *Proc. DL'07*, pages 451–458. CEUR-WS.org, 2007.

[34] Ognjen Savković and Diego Calvanese. Introducing datatypes in *DL-Lite*. In *Proc. ECAI'12*, pages 720–725, 2012.

[35] Klaus Schild. A correspondence theory for terminological logics: Preliminary report. In John Mylopoulos and Raymond Reiter, editors, *Proc. IJCAI'91*, pages 466–471. Morgan Kaufmann, 1991.

[36] Sarvjeet Singh, Chris Mayfield, Sagar Mittal, Sunil Prabhakar, Susanne Hambrusch, and Rahul Shah. Orion 2.0: native support for uncertain data. In *Proc. SIGMOD'08*, pages 1239–1242. ACM, 2008.

[37] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Autonomous Robots*, 5(3-4):253–271, 1998.

[38] Seinosuke Toda and Osamu Watanabe. Polynomial-time 1-Turing reductions from #PH to #P. *Theoretical Computer Science*, 100(1):205–221, 1992.

[39] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM computing surveys (CSUR)*, 38(4):13, 2006.

# A Classical Query Entailment

## A.1 Query Entailment in $\mathcal{EL}(\mathsf{R}_{>})$

We investigate the complexity of query entailment in classical $\mathcal{EL}(\mathsf{R}_{>})$ KBs. For this, we show that query entailment in $\mathcal{EL}(\mathsf{R}_{>})$ can be reduced to query entailment in $\mathcal{EL}$ using a polynomial transformation, so that known complexity results for query entailment in $\mathcal{EL}$ can be transferred. Note that $\mathcal{EL}(\mathsf{R}_{>})$ concepts contain no abstract features, and therefore also no feature chains. Hence, concrete domain predicates only occur in subconcepts of the form $\exists g.{>}_n$. The transformation applies to both the KB and the query, and proceeds as follows, where concept names $E_{g>n}$ are fresh.

1. Replace every subconcept $\exists g.{>}_n$ by $E_{g>n}$.
2. For every $E_{g>n}$ and $E_{g>m}$ introduced this way s.t. $n > m$, add the GCI $E_{r>n} \sqsubseteq E_{r>m}$.

Let $\mathcal{K}$ be an $\mathcal{EL}(\mathsf{R}_{>})$ KB and $q$ a query, and let $\mathcal{K}'$ and $q'$ be the $\mathcal{EL}$ KB and the query resulting from the above transformation. To see that $\mathcal{K} \not\models q$ implies $\mathcal{K}' \not\models q$, one easily establishes that, by interpreting the fresh concepts appropriately, every model $\mathcal{I}$ of $\mathcal{K}$ can be extended to a model $\mathcal{I}'$ of $\mathcal{K}'$ such that $\mathcal{I}' \models q'$ iff $\mathcal{I} \models q$. To show that also $\mathcal{K}' \not\models q$ implies $\mathcal{K} \not\models q$, let $\mathcal{I}'$ be a model of $\mathcal{K}'$. Let $n_1, ..., n_m$ be the numbers occurring in number restrictions in $\mathcal{K}$ such that $n_i < n_{i+1}$ for all $1 \leq i \leq m$. Now, process all pairs $(x, E_{g>n_i})$ of domain elements $x \in \Delta^{\mathcal{I}'}$ and introduced concepts $E_{g>n_i}$ such that (i) $x \in (E_{g>n_i})^{\mathcal{I}'}$ and (ii) there is no $j > i$ with $x \in (E_{g>n_j})^{\mathcal{I}'}$. For each such pair, set $g^{\mathcal{I}}(x) = \frac{n_i + n_{i+1}}{2}$ if $i < m$, and set $g^{\mathcal{I}}(x) = n_i + 1$ otherwise. One easily establishes that $\mathcal{I}$ is a model of $\mathcal{K}$, and that $\mathcal{I} \models q$ iff $\mathcal{I}' \models q'$. We obtain that $\mathcal{K} \not\models q$ iff $\mathcal{K}' \not\models q'$, and therefore that query entailment in $\mathcal{EL}(\mathsf{R}_{>})$ can be polynomially reduced to query entailment in $\mathcal{EL}$.

Based on the results in [5, 33] for respectively axiom entailment and UCQ entailment in $\mathcal{EL}$ KBs, we thus obtain the following completeness results.

**Theorem 1.** *Deciding entailments of AQs and UCQs in classical $\mathcal{EL}(\mathsf{R}_{>})$ KBs is* P*-complete in data complexity, and respectively* P*- and* NP*-complete in combined complexity.*

## A.2 Query Entailment in $\mathcal{ALC}(\mathsf{R})$

Due to the feature chains, such a direct reduction is not possible for $\mathcal{ALC}(\mathsf{R})$. Instead, the procedure presented in [29] for answering CQs in $\mathcal{SHQ}$ KBs can be used to decide query entailment. This method exploits the fact that $\mathcal{SHQ}$ has the forest model property, which allows to decide query entailment by deciding consistency of a series of extensions of the original knowledge base by so-called *spoilers*, which are constructed based on different rewritings and partitionings of the query. When each extension is unsatisfiable, the query is entailed. The spoilers have to be formulated in the DL $\mathcal{SHQ}^{\cap}$, which extends $\mathcal{SHQ}$ by role conjunctions, since different roles can connect to the same individual in a query. As the size of each spoiler is polynomially bounded, and there are at most exponentially many spoiler, the authors obtain an ExpTime upper bound on CQ-entailment w.r.t. combined complexity. Note that in order to decide whether a query is not entailed, it is sufficient to non-deterministically guess one such spoiler, for which satisfiability can be checked in NP data complexity, so that we obtain a data complexity of coNP for query entailment. Regarding UCQs, we note that $\mathcal{K} \not\models q_1 \vee q_2$ iff $\mathcal{K} \not\models q_1$ and $\mathcal{K} \not\models q_2$. We can therefore decide whether a UCQ is not entailed using an NP procedure that guesses one spoiler for each conjunct of the UCQ one after another, so that

UCQ entailment remains in coNP w.r.t data complexity and in ExpTime w.r.t. combined complexity.

While, due to the concrete domain, $\mathcal{ALC}(\mathsf{R})$ is not a sublanguage of $\mathcal{SHQ}$, $\mathcal{ALC}(\mathsf{R})$ also has the forest-tree model property. Since furthermore according to our definition of queries concrete features are only referred to within concepts in a query, the method can be used without modifications to decide query entailment for $\mathcal{ALC}(\mathsf{R})$ knowledge bases. However, for this we need to decide satisfiability of KBs that use role conjunctions. It can shown using little modifications of the proofs in [25] for ExpTime-completeness of $\mathcal{SHIQ}(\mathsf{R})$ TBox satisfiability, and the proofs in [26, Section 6.2] for ExpTime-completeness of a less expressive version of $\mathcal{ALC}(\mathsf{R})$, that satisfiability of $\mathcal{ALC}(\mathsf{R})^{\cap}$ KBs can be decided in ExpTime combined and NP data complexity, which are the same complexities as for $\mathcal{SHQ}$. The details of these adaptations given in the following two subsections. We thus have the following result.

**Theorem 2.** *UCQ-entailment in $\mathcal{ALC}(\mathsf{R})$ knowledge bases is* coNP*-complete in data complexity and* ExpTime*-complete in combined complexity.*

## A.3 Satisfiability of $\mathcal{ALC}(\mathsf{R})^{\cap}$ TBoxes

For the logic $\mathcal{SHIQ}(\mathsf{R})$, it is shown in [25] that TBox satisfiability is in ExpTime. More precisely, the authors consider a slightly more expressive extension with the numerical domain, which also allows for expressions of the form $\forall r g_1, g_2.P$ and $\exists r g_1, g_2.P$, where $P$ is a concrete domain predicate, $r$ can be any role and $g_1, g_2 \in N_{cF}$. Note that longer paths are still only allowed to be used in existential restrictions and with only abstract features as roles. To make the following simpler, we assume that the same constructs are allowed in $\mathcal{ALC}(\mathsf{R})$. In addition, we extend our logic by role conjunctions defined next.

We denote by $\mathcal{ALC}(\mathsf{R})^{\cap}$ the extension of $\mathcal{ALC}(\mathsf{R})$ with *role conjunctions*, that is, we allow role expressions of the form $r_1 \cap \ldots \cap r_n$, $n > 0$, where $r_1, \ldots, r_n \in N_r$. Role expressions can occur anywhere in $\mathcal{ALC}(\mathsf{R})^{\cap}$ where roles can occur in $\mathcal{ALC}(\mathsf{R})$. Given a set $\mathbf{R}$ of role names $\{r_1, \ldots, r_n\}$, we may abbreviate the role conjunction $r_1 \cap \ldots \cap r_n$ by $\bigcap \mathbf{R}$.

We present a decision procedure for $\mathcal{ALC}(\mathsf{R})^{\cap}$ that very closely follows the one presented in [25]. This reasoning procedure assumes TBoxes to consist of a single concept inclusion of the form $\top \sqsubseteq C_{\mathcal{T}}$, where $C_{\mathcal{T}}$ is in a normal form defined next. A concept $C$ is in *negation normal form* (NNF), if every negation symbol in $C$ occurs in front of a concept name. $C$ is in *path normal form* (PNF) if

1. it is in negation normal form,
2. for all subconcepts of the form $\exists u.\oplus_r$, $\oplus \in \{<, =, >\}$, we have $u \in N_{cF}$, and
3. for all subconcepts of the forms $\exists (u, v).\oplus$ and $\forall (u, v).\oplus$, $\oplus \in \{<, =, >\}$, $v$ is of the form $g_2$, $g_2 \in N_{cF}$, and $u$ is either of the form $g_1$ or $Rg_1$, where $g_1 \in N_{cF}$ and $R$ is a role name or a role conjunction.

A TBox is in PNF if it is of the form $\top \sqsubseteq C_{\mathcal{T}}$, where $C_{\mathcal{T}}$ is in PNF.

As shown in [25], every TBox can be polynomially transformed into PNF using a structural transformation on paths, that is, by introducing fresh concrete features that are defined to be equivalent to their path counter parts. The transformed TBox in PNF is polynomial in size of the input TBox. In the following, we assume all TBoxes $\mathcal{T}$ to be in PNF.

The reasoning procedure is based on an abstraction of models called Hintikka trees, which we now introduce step by step.

For a concept $C$, we denote by $\overline{C}$ the NNF of $\neg C$. We define the *concept closure* $\mathrm{ccl}(\mathcal{T})$ of a TBox $\mathcal{T}$ as $\{C, \overline{C} \mid C \in sub(\mathcal{T})\} \cup \{\top\}$. A *Hintikka set $t$ for $\mathcal{T} = \{\top \sqsubseteq C_\mathcal{T}\}$* is any subset $t \subseteq \mathrm{ccl}(\mathcal{T})$. A Hintikka set $t$ is *consistent with $\mathcal{T}$* if it satisfies the following:

- $C_\mathcal{T} \in t$,
- $\neg A \in t$ iff $A \notin t$ for all $A \in \mathrm{ccl}(\mathcal{T}) \cap N_c$,
- $C \sqcap D \in t$ only if $C, D \in t$,
- $C \sqcup D \in t$ only if $C \in t$ or $D \in t$, and
- $\top \in t$.

A *constraint system* is a set of inequations of the forms $x \oplus r$ and $x \oplus y$, where $x, y$ are variables, $r \in \mathbb{R}$ and $\oplus \in \{<, =, >\}$. The *solution of a constraint system $\mathcal{E}$* is a mapping from variables to real numbers such that every inequation in $\mathcal{E}$ is satisfied. A *Hintikka label for $\mathcal{T}$* is a tuple $(t, \omega, \mathcal{E})$, where $t$ is a Hintikka set for $\mathcal{T}$, $\omega \subseteq N_r$ and $\mathcal{E}$ a constraint system that contains the following inequations:

- $x_g \oplus r$ for every $\exists g.\oplus_r \in t$, $\oplus \in \{<, =, >\}$,
- $x_{g_1} \oplus x_{g_2}$ for every $\exists (g_1, g_2).\oplus \in t$, $\oplus \in \{<, =, >\}$,
- at least one inequation of the form $x_{g_1} \oplus x_{g_2}$, $\oplus \in \{<, =, >\}$ for every pair of variables $x_{g_1}$, $x_{g_2}$ occurring in $\mathcal{E}$, and
- At least one inequation of the form $x_g \oplus r$, $\oplus \in \{<, =, >\}$, for every variable $x_g$ in $\mathcal{E}$ and every number $r$ that occurs in a concept $\exists d.\otimes_r \in \mathrm{sub}(\mathcal{T})$,$\otimes \in \{<, =, >\}$.

A Hintikka label $(t, \omega, \mathcal{E})$ for $\mathcal{T}$ is consistent iff $t$ is consistent with $\mathcal{T}$ and $\mathcal{E}$ has a solution.

Hintikka labels represent domain elements and their concrete features in an interpretation. The set $\omega$ identifies the incoming role edges to these domain elements. The more complex interactions with other domain elements are captured by an abstraction called Hintikka tuples.

**Definition 5.** Let $b_\mathcal{T}$ be the number of subconcepts of the form $\exists R.C$ and $\exists R g_1, g_2.\oplus$ occurring in $\mathrm{ccl}(C_\mathcal{T})$: $b_\mathcal{T} = \|\{C \in \mathrm{sub}(C_\mathcal{T}) \mid C = \exists R.D \text{ or } C = \exists R g_1, g_2.\oplus\}\|$. A *Hintikka tuple $(T_0, \ldots, T_{b_\mathcal{T}})$ for $\mathcal{T}$* is a $b_\mathcal{T} + 1$-tuple of Hintikka labels $T_i = (t_i, \omega_i, \mathcal{E}_i)$ for $\mathcal{T}$ such that

- If $\exists \bigcap \mathbf{R}.C \in t_0$, then there exists an index $i$, $0 < i \leq b_\mathcal{T}$, such that $\mathbf{R} \subseteq \omega_i$ and $C \in t_i$.
- If $\forall \bigcap \mathbf{R}.C \in t_0$, then for all $i$, $0 < i \leq b_\mathcal{T}$, such that $\mathbf{R} \subseteq \omega_i$, also $C \in t_i$.
- For all $s \in N_{aF}$, there is at most one $i$, $0 < i \leq b_\mathcal{T}$, such that $s \in \omega_i$.

The *constraint system induced by a Hintikka tuple* consists of the inequations in $\mathcal{E}_0$, together with the following inequations:

1. $x_{ig} \oplus r$ for every $x_g \oplus r \in \mathcal{E}_i$ and $0 < i \leq b_\mathcal{T}$
2. $x_{ig_1} \oplus x_{ig_2}$ for every $x_{g_1} \oplus x_{g_2} \in \mathcal{E}_i$ and $0 < i \leq b_\mathcal{T}$.
3. $x_{ig_1} \oplus x_{g_2}$, for every $\exists \bigcap \mathbf{R} g_1, g_2.\oplus \in t_0$ and some $i$ with $0 < i \leq b_\mathcal{T}$ and $\mathbf{R} \subseteq \omega_i$.
4. $x_{ig_1} \oplus x_{g_2}$, for every $\forall \bigcap \mathbf{R} g_1, g_2.\oplus \in \mathcal{C}$ and $i$ such that $0 < i \leq b_\mathcal{T}$ and $\mathbf{R} \subseteq \omega_i$.

A Hintikka tuple is *consistent* iff the Hintikka set in every Hintikka label is consistent and the constraint system induced by the Hintikka tuple has a solution.

Intuitively, consistent Hintikka tuples represent model fragments that contain a root domain element and its successors. The sets $\omega_i$ contain the roles that lead from the root domain element, represented by $T_0$, to the domain elements represented by $T_i$. This way, Hintikka tuples contain a labelling of roles to successors. A Hintikka tuple does not necessarily represent a situation where a domain element is connected to $b_\mathcal{T}$ other domain elements, because there may be labels $T_i = (t_i, \omega_i, \mathcal{E}_i)$ in the tuple for which $\omega_i = \emptyset$.

A *Hintikka tree* for $\mathcal{T}$ is a $b_\mathcal{T}$-ary tree $H = (V, E)$ such that $V$ is a set of Hintikka labels and for each $v \in V$ and its direct successors $\{v_i \mid (v, v_i) \in E\}$, the tuple $(v, v_0, \dots v_{b_\mathcal{T}-1})$ is a consistent Hintikka tuple for $\mathcal{T}$. Note that, even though $H$ does not have labelled edges, we can obtain such a labeling based on the role sets $\omega_i$ in each Hintikka label.

Given a Hintikka tree $(V, E)$, we can obtain a corresponding constraint system by combining the constraint systems induced by all Hintikka tuples and renaming the variables accordingly. As shown in [25] for Hintikka trees without role conjunctions, if all Hintikka tuples are consistent, this constraint system always has a solution. Clearly role conjunctions have no impact on this result, so that it also holds for our case. Therefore, it is always possible to construct a model for $\mathcal{T}$ based on a Hintikka tree for $\mathcal{T}$. It follows that $\mathcal{T}$ is satisfiable iff $\mathcal{T}$ has a Hintikka tree.

In order to determine satisfiability of $\mathcal{ALC}(\mathsf{R})$ TBoxes, it suffices to determine whether there exists a Hintikka tree for it. The existence of such a tree can be decided using type elimination. For this, we use an abstraction of Hintikka trees that we call Hintikka structure.

**Definition 6.** A *Hintikka structure* for $\mathcal{T}$ is a set $H$ of consistent Hintikka tuples for $\mathcal{T}$ such that for every tuple $(T_0, \dots T_{b_\mathcal{T}}) \in H$ and every $i \leq b_\mathcal{T}$, there exists a Hintikka tuple $(T'_0, \dots T'_{b_\mathcal{T}}) \in H$ such that $T_i = T'_0$.

It is easy to verify that a Hintikka tree exists for a TBox $\mathcal{T}$ iff there exists a non-empty Hintikka structure for $\mathcal{T}$. In order to determine whether $\mathcal{T}$ is satisfiable, it is sufficient to check whether there exists a Hintikka structure for $\mathcal{T}$. We first argue that the number of Hintikka tuples for $\mathcal{T}$ is exponentially bounded in the size of $\mathcal{T}$. The size of $\mathrm{ccl}(\mathcal{T})$ is linear in the size of $\mathcal{T}$ and so is $b_\mathcal{T}$. The number of different Hintikka sets for $\mathcal{T}$ is exponentially bounded, because it is restricted to the subsets of $\mathrm{ccl}(\mathcal{T})$ and the number of combinations of concrete features and numbers that occur explicitly in $\mathcal{T}$. Since $b_\mathcal{T}$ is linear in the size of $\mathcal{T}$, the number of Hintikka tuples for $\mathcal{T}$ is exponentially bounded as well.

The existence of a Hintikka structure can now be decided using the following ExpTime type elimination procedure.

1. Construct the set **H** of all Hintikka tuples for $\mathcal{T}$
2. While possible, remove from **H** a Hintikka tuple that serves as a counter example for **H** being a Hintikka structure.
3. If the procedure terminates with an empty set, return "Unsatisfiable", otherwise, return "Satisfiable".

The second step requires time polynomial in $|\mathbf{H}|$ per iteration, since deciding consistency of a constraint system is in P, and we have to check at most $|\mathbf{H}|$ other Hintikka tuples in this step. Because in every iteration step, we remove one Hintikka tuple, and **H** contains at most exponentially many elements, the complete procedure runs in exponential time.

**Theorem 3.** *Deciding satisfiability of $\mathcal{ALC}(\mathsf{R})^\cap$ TBoxes is* ExpTime*-complete.*

As a side-result, we consider a special case of concept satisfiability with respect to a TBox, which will come in handy in the next section.

**Lemma 3.** *Let $C$ be a concept of the form $A_1 \sqcap \ldots \sqcap A_n \sqcap C'$, where $A_1, \ldots, A_n \in N_c$ and $C'$ is any $\mathcal{ALC}(\mathsf{R})^\cap$ concept, and let $\mathcal{T}$ be an $\mathcal{ALC}(\mathsf{R})^\cap$ ontology. Then, whether $C'$ is satisfiable w.r.t. $\mathcal{T}$ can be decided in time exponential in $|C| + |\mathcal{T}|$, and polynomial in $n$.*

*Proof.* We can adapt the above algorithm such that it starts from a different set of Hintikka sets. Namely, we consider only the Hintikka sets $\{t, t \cup \{A_1, \ldots, A_n\} \mid t \in \mathrm{ccl}(C') \cup \mathrm{ccl}(\mathcal{T})\}$ when constructing the initial set of Hintikka labels. Note that this set of types is twice as big as the set of types we would usually consider, so that its size is independent of $n$. We then perform the algorithm as before and return "satisfiable" iff the method terminates with a Hintikka structure that contains a Hintikka label with a Hintikka set $t \supseteq \{A_1, \ldots, A_n, C'\}$. The number of Hintikka labels, and consequently of elimination steps taken, is exponential in $|C| + |\mathcal{T}|$, but independent on $n$, while the time required in each step is polynomial in $n$. Hence the algorithm runs in time polynomial in $n$. It is sound, since in the case of a successful run, we can construct a model $\mathcal{I}$ from the Hintikka structure such that there is a domain element $a \in \Delta^\mathcal{I}$ with $a \in (A_1 \sqcap \ldots \sqcap A_n \sqcap C')^\mathcal{I}$. For completeness, assume we have a model $\mathcal{I}$ of $\mathcal{T}$ with $a \in C^\mathcal{I}$ for some $a \in \Delta^\mathcal{I}$. Then, there must be a consistent Hintikka set $t$ such that $C', A_1, \ldots, A_n \in t$ and $a \in D^\mathcal{I}$ for every $D \in t$. Starting from $a$, we can inductively construct a Hintikka tree, which correspondingly implies the existence of a Hintikka structure obtainable by the presented algorithm. □

## A.4 Satisfiability of $\mathcal{ALC}(\mathsf{R})^\cap$ Knowledge Bases

To show that satisfiability of $\mathcal{ALC}(\mathsf{R})^\cap$ KBs can also be decided in ExpTime combined complexity, and in NP w.r.t. data complexity, we modify the decision procedure from [26, Section 6.2] for $\mathcal{ALC}(\mathsf{R})$ KBs without unary domain predicates to support full $\mathcal{ALC}(\mathsf{R})^\cap$.

The method from [26, Section 6.2] uses a tableau to nondeterministically construct an ABox precompletion, on which it then uses a decision procedure for $\mathcal{ALC}(\mathsf{R})$ concept satisfiability, where again the input is asssumed to be in PNF. paths.

The precompletions generated by the algorithm use in addition to the assertions introduced in the preliminaries also assertions of the form $C(a)$, where $C$ can be a complex concept, and make use of variables for concrete feature values. These variables occur in assertions of the form $g(a, x)$, $g \in N_{cF}$, $x \in N_v$, and in statements of the form $x_1 \oplus x_2$, $x_1, x_2 \in N_v$, $\oplus \in \{<, =, >\}$ with the obvious semantics. In order to extend the approach for unary predicates, we further allow assertions of the form $x \oplus r$, where $\oplus \in \{<, =, >\}$ and $r \in \mathbb{R}$.

The precompletion calculus is show in Figure 2, in which we marked rules that differ to the original calculus with an asterisk. All non-deterministic steps are don't know non-deterministic steps, while the order in which rules are applied is don't care non-determinism. The rules **R**$\exists f$ and **R**$\forall$ are generalisations of corresponding rules from [26] for role conjunctions, while **R**$\exists f$**'** takes additional care of role conjunctions in existential role restrictions involving abstract features. The rules **Rc1'**, **Rc2'** and **Rch'** are the unary-predicate versions of **Rc1**, **Rc1** and **Rch**.

For $a \in N_i$ and precompletion ABox $\mathcal{A}$, use $\mathit{con}(\mathcal{A}, a)$ to denote the *reduction concept* $\bigsqcap_{C(a) \in \mathcal{A}} C$. A precompletion ABox $\mathcal{A}$ has a clash if for some $a \in N_i$, $\mathit{con}(\mathcal{A}, a)$ is unsatisfiable in $\mathcal{T}$, or if the set of inequations $x_1 \oplus x_2$, $x \oplus r \in \mathcal{A}$ form an unsatisfiable constraint system. The algorithm returns "satisfiable" if there exists a precompletion ABox saturated by the calculus that does not contain a clash, and otherwise it returns "unsatisfiable".

The rules **Rch** and **Rch'** ensure that for each individual and each concrete feature, the re-

**R⊓** If $(C_1 \sqcap C_2)(a) \in \mathcal{A}$ and $C_1(a), C_2(a) \notin \mathcal{A}$, then $\mathcal{A} := \mathcal{A} \cup \{C_1(a), C_2(a)\}$.

**R⊔** If $(C_1 \sqcup C_2)(a) \in \mathcal{A}$ and $\{C_1(a), C_2(a)\} \cap \mathcal{A} = \emptyset$, then $\mathcal{A} := \mathcal{A} \cup \{C_i(a)\}$ for some non-deterministically picked $i \in \{1, 2\}$.

⋆ **R∃$f$** If $(\exists \bigcap \mathbf{R}.C)(a) \in \mathcal{A}$, $r \in \mathbf{R} \cap N_{aF}$ and $r(a, b) \in \mathcal{A}$, set $\mathcal{A} := \mathcal{A} \cup \{C(b)\}$.

⋆ **R∃$f$'** If $(\exists \cap \mathbf{R}.C)(a) \in \mathcal{A}$, $r \in \mathbf{R} \cap N_{aF}$, $s \in \mathbf{R}$, $r(a, b) \in \mathcal{A}$ and $s(a, b) \notin \mathcal{A}$, then $\mathcal{A} := \mathcal{A} \cup \{s(a, b)\}$.

⋆ **R∀** If $(\forall \bigcap \mathbf{R}.C)(a) \in \mathcal{A}$ and $r(a, b) \in \mathcal{A}$ for all $r \in \mathbf{R}$, then $\mathcal{A} := \mathcal{A} \cup \{C(b)\}$.

**Rc1** If $(\exists g_1, g_2.\oplus)(a) \in \mathcal{A}$, $g_i(a, x_i) \in \mathcal{A}$ for $i \in \{1, 2\}$ and $x_1 \oplus x_2 \notin \mathcal{A}$, then $\mathcal{A} := \mathcal{A} \cup \{x_1 \oplus x_2\}$.

**Rc2** If $(\exists f g_1, g_2.\oplus)(a), f(a, b) \in \mathcal{A}$ and there are no $x_1, x_2 \in N_v$ s.t.

- $g_1(b, x_1) \in \mathcal{A}$,
- $g_2(a, x_2) \in \mathcal{A}$, and
- $x_1 \oplus x_2 \in \mathcal{A}$,

then $\mathcal{A} := \mathcal{A} \cup \{g_1(b, x_1), g_2(a, x_2), x_1 \oplus x_2\}$, where $x_1$ and $x_2$ are fresh.

**Rc3** Symmetric to **Rc2** but for $(\exists g_1, f g_2.\oplus)(a) \in \mathcal{A}$.

**Rch** If $g_1(a, x_1), g_2(a, x_2) \in \mathcal{A}$ and $\{(\exists g_1, g_2.<, \exists g_1, g_2.=, \exists g_1, g_2.>\} \cap \mathcal{A} = \emptyset$, then $\mathcal{A} := \mathcal{A} \cup \{\exists g_1, g_2.\oplus\}$ for some non-deterministically picked $\oplus \in \{<, =, >\}$.

⋆ **Rc1'** If $\exists g.\oplus_r(a) \in \mathcal{A}$ and and $g(a, x) \in \mathcal{A}$, then $\mathcal{A} := \mathcal{A} \cup \{x \oplus r\}$.

⋆ **Rc2'** If $\exists g.\oplus_r(a) \in \mathcal{A}$ and there is no $y \in N_v$ with $g(a, y) \in \mathcal{A}$, then $\mathcal{A} := \mathcal{A} \cup \{g(a, x), x \oplus r\}$, where $x$ is fresh.

⋆ **Rch'** If $g(a, x) \in \mathcal{A}$, $r \in \mathbb{R}$ occurs in $\mathcal{T}$ and $\{(\exists g.>_r)(a), (\exists g.=_r)(a), (\exists g.<_r)(a)\} \cap \mathcal{A} = \emptyset$, then $\mathcal{A} := \mathcal{A} \cup \{(\exists g.\oplus_r)(a)$ for some non-deterministically picked $\oplus \in \{<, =, >\}$.

**R$\mathcal{T}$** If $C \sqsubseteq D \in \mathcal{T}$ and $(\neg C \sqcup D)(a) \notin \mathcal{A}$, then $\mathcal{A} := \mathcal{A} \cup \{(\neg C \sqcup D)(a)\}$.

**Rfe** If $f(a, b), f(a, c) \in \mathcal{A}$ for $b \neq c$ and $f \in N_{aF}$, replace $b$ in $\mathcal{A}$ by $c$

**Rge** If $g(a, x_1), g(a, x_2) \in \mathcal{A}$ for $x_1 \neq x_2$ and $g \in N_{cF}$, replace $x_1$ in $\mathcal{A}$ by $x_2$

Figure 2: ABox precompletion calculus for $\mathcal{ALC}(\mathsf{R})$.

lationships to any number occurring in the TBox, as well as to other concrete features, are determined. This makes sure that, if $con(\mathcal{A}, a)$ has a model, the values of the concrete features of $a$ are within the bounds that occur in relation with other individuals in the ABox, so that we can "plug in" theses models in order to build a model based on a precompletion ABox.

One easily verifies that the modified rules do not affect the complexity of the algorithm, so that it still runs in ExpTime w.r.t combined complexity. Concerning data complexity, recall that our input only contains concepts which are concept names. One easily verifies that the size of any precompletion ABox is polynomially bounded by the size of the input ABox [26, Lemma 6.24]. Inspection of the rules shows that the only elements in $con(\mathcal{A}, a)$ that are not introduced due to the TBox are concept names $A$ for which $A(a) \in \mathcal{A}$, as well as concepts of the form $\exists g_1, g_2.\oplus$ and $\exists g.\oplus_r$, $\oplus \in \{<, =, >\}$. Therefore, $con(\mathcal{A}, a)$ is of the form $C_1 \sqcap \ldots \sqcap C_n \sqcap C'$, where each $C_i$ does not contain nested concepts, and $n$ is linearly bounded by $\mathcal{A}$. Using Lemma 3, it follows that the satisfiability of each reduction concept can be decided in time polynomial in the size of the ABox, though in exponential time w.r.t. to the complete KB. We obtain that the algorithm runs in NP w.r.t. data complexity.

**Theorem 4.** *Satisfiability of $\mathcal{ALC}(\mathsf{R})^{\cap}$ KBs is decidable in* ExpTime *w.r.t. combined complexity and in* NP *w.r.t. data complexity.*

# B Probabilistic Query Entailment

## B.1 Semantics of Probabilistic Knowledge Bases

We show that the definition of probability measure spaces for probabilistic ABoxes provided in Section 3.1 is indeed a definition, by showing that the measure function $\mu_{\mathcal{A}}$ is uniquely defined for every ABox $\mathcal{A}$.

**Lemma 4.** *For every probabilistic ABox $\mathcal{A}$, the measure function $\mu_{\mathcal{A}}$ is uniquely defined.*

*Proof.* Let $\mathcal{A}$ be a probabilistic knowledge base, and let $M^1_{\mathcal{A}} = (\Omega_{\mathcal{A}}, \Sigma_{\mathcal{A}}, \mu_1)$ and $M^2_{\mathcal{A}} = (\Omega_{\mathcal{A}}, \Sigma_{\mathcal{A}}, \mu_2)$ be two probability measure spaces for $\mathcal{A}$ that satisfy the conditions given in Section 3.1. We show that then, for all $W \in \Sigma_{\mathcal{A}}$, $\mu_1(W) = \mu_2(W)$. One can easily verify that for sets $A \in \Sigma_{\mathcal{A}}$ that satisfy Conditions 1–3 in the definition of event spaces over $\Omega_{\mathcal{A}}$, as well as any set that can be obtained by countably many set operations on these, $\mu_1(A)$ and $\mu_2(A)$ are uniquely determined by these conditions. Specifically, for all sets of possible worlds $W \in \Sigma_{\mathcal{A}}$ satisfying $g(a) \in [x, y]$ for some $g \in N_{cF}, a \in N_i, x, y \in \mathbb{R} \cup \{-\infty, \infty\}$, and all sets that can be obtained using countable set operations on these, we have $\mu_1(W) = \mu_2(W)$. It remains to show that also for sets $W \in \Sigma_{\mathcal{A}}$ of possible worlds that satisfy Condition 4, that is, that satisfy $g_1(a) < g_2(b)$, $g_1, g_2 \in N_{cF}, a, b \in N_i$, we have $\mu_1(W) = \mu_2(W)$

**Claim 1.** *For all $g_1(a) \sim f_1, g_2(b) \sim f_2 \in \mathcal{A}$, $\mu_1(g_1(a) < g_2(b)) = \mu_2(g_1(a) < g_2(b))$.*

*Proof of claim.* Let $A = \{w \in \Omega_{\mathcal{A}} \mid w \models g_1(a) < g_2(b)\}$. We have to show $\mu_1(A) = \mu_2(A)$. The proof is by contradiction. Assume $\mu_1(A) \neq \mu_2(A)$. Then, there exists $n \in \mathbb{N}$ such that $|\mu_1(A) - \mu_2(A)| > 2^{-n}$. We can divide $\Omega_{\mathcal{A}}$ into $2^n$ subsets of equal probability $2^{-n}$ by partitioning the range of $g(a)$. Namely, there is a sequence $x_1, \ldots, x_{2^n-1}$ of real numbers such that

$$\begin{aligned}\mu_1(g_1(a) \leq x_1) &= \mu_1(g_1(a) \in (x_i, x_{i+1}]) \\ &= \mu_1(g_1(a) > x_{2^n-1}) \\ &= 2^{-n}\end{aligned}$$

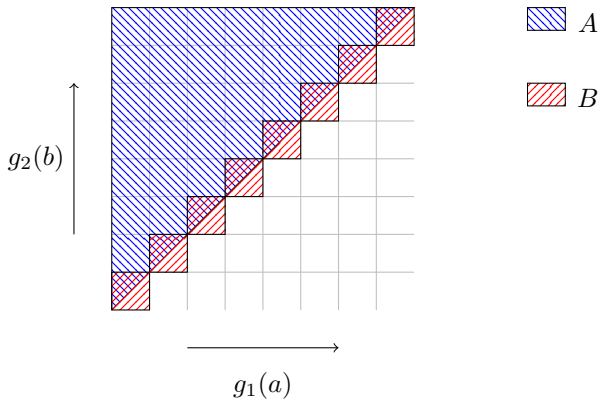


Figure 3: Illustration of the sets $A$ and $B$ in the proof for Lemma 4.

for all $0 < i < 2^n - 1$. Similarly, there is a sequence $y_1, \ldots, y_{2^n-1}$ of real numbers such that

$$\begin{aligned}\mu_1(g_2(b) \leq y_1) &= \mu_1(g_2(b) \in (y_i, y_{i+1}]) \\ &= \mu_1(g_2(b) > x_{2^n-1}) \\ &= 2^{-n}\end{aligned}$$

for all $0 < i < 2^n - 1$.

Given a set of statements $\alpha_1, \ldots, \alpha_m$, let $W(\alpha_1, \ldots \alpha_m) = \{w \in \Omega_{\mathcal{A}} \mid w \models \alpha_1, \ldots, \alpha_m\}$. We define a set $B \subseteq \Omega_{\mathcal{A}}$ of possible worlds as follows:

$$\begin{aligned}B = {} & W(g_1(a) < x_1, g_2(b) < y_1) \\ \cup & \bigcup_{\substack{1<i,j<2^{n-1}, \\ [x_i,x_{i+1}]\cap[y_j,y_{j+1}]\neq\emptyset}} W(g_1(a) \in [x_i, x_{i+1}], g_2(b) \in [y_j, y_{j+1}]) \\ \cup & W(g_1(a) > x_{2^{n+1}-1}, g_2(b) > x_{2^{n+1}-1}).\end{aligned}$$

The sets $A$ and $B$ are illustrated in Figure B.1. Clearly, $B \in \Sigma_{\mathcal{A}}$. It is further easy to verify that each square in the figure that is a part of $B$ has an assigned probability of $(2^{-n})^2$ for both measure functions. We therefore obtain that that $\mu_1(B) = \mu_2(B) = 2^n \cdot (2^{-n})^2 = 2^{-n}$. If we consider the set $A \setminus B$, we see that this set can be obtained by countable set operations on elements in $\Sigma_{\mathcal{A}}$ that satisfy Conditions 1–3 (that is, without using $A$). As observed at the beginning of this proof, this implies that $\mu_1(A \setminus B) = \mu_2(A \setminus B)$. From this, we obtain the following bounds on the difference between $\mu_1(A)$ and $\mu_2(A)$:

$$\begin{aligned}|\mu_1(A) - \mu_1(A)| &= |(\mu_1(A \setminus B) + \mu_1(A \cap B)) - (\mu_2(A \setminus B) + \mu_2(A \cap B))| \\ &= |\mu_1(A \cap B) - \mu_2(A \cap B)| \\ &\leq \max(\mu_1(A \cap B), \mu_2(A \cap B)) \\ &\leq \max(\mu_1(B), \mu_2(B)) \\ &= \mu_1(B) \\ &= 2^{-n}\end{aligned}$$

We have $|\mu_1(A) - \mu_2(A)| \leq 2^{-n}$, which contradicts $|\mu_1(A) - \mu_2(A)| > 2^{-n}$. Hence, there cannot be any natural number $n \in \mathbb{N}$ such that $|\mu_1(A) - \mu_2(A)| > 2^{-n}$, so that we have $\mu_1(A) = \mu_2(A)$. ■

It follows that for any set $A$ satisfying Conditions 1–5 in the definition of the event space $\Sigma_{\mathcal{A}}$ over $\mathcal{A}$, we have $\mu_1(A) = \mu_2(A)$, so that the probability measure space for $\mathcal{A}$ is uniquely defined. □

## B.2 Feasibility Conditions for PDFs

We show that the set $\mathcal{P}_{\mathsf{ex}}$ of pdfs from Example 3 is #·P-feasible. For this, we first show that every function in $\mathcal{P}_{\mathsf{ex}}$ is P-computable.

**Lemma 5.** *Every function in $\mathcal{P}_{\mathsf{ex}}$ is P-computable.*

*Proof.* The functions in $\mathcal{P}_{\mathsf{ex}}$ use as constants rational numbers, as well as the real constants $e$ and $\pi$, and are further composed of the following basic operations: square-root, subtraction, multiplication, division by a constant and exponentiation. We recall feasibility results for the involved real numbers and basic computations from [8, 21]. $e$ and $\pi$, square-root, subtraction and multiplication are all P-computable. Furthermore, exponentiation $e^x$ is P-computable if $x$ is restricted to negative numbers. The function $f : x \mapsto \frac{1}{x}$ is not P-computable, however, given any two numbers $x, y \in \mathbb{D}$ in binary representation, $\frac{x}{y}$ can be approximated in time polynomial in the number of bits in $x$ and $y$ [8]. Using the definition of P-computability of real functions using function oracles, one can easily establish that any function that can be composed of a constant number of P-computable functions and constants is P itself. We can thus establish that every function in $\mathcal{P}$ is P-computable:

1. $\mathsf{uniform}(a, b) : [a, b] \to \mathbb{R}^+$ only uses division by constants, which is P-computable.
2. $\mathsf{exp}(\lambda) : [0, \infty] \to \mathbb{R}^+$ uses multiplication by constants and exponentiation with a negative value, and is thus P-computable.
3. $\mathsf{norm} : \mathbb{R} \to \mathbb{R}^+$ uses the real constants $e$ and $\pi$, subtraction, division by constants, multiplication and exponentiation. The exponential has as exponent $-(x-\mu)^2/2\omega$, which is always negative, so that again we obtain P-computability.

Note further that in each case, the complexity is polynomial not only in $n$, but also in the constants involved. We obtain that all functions in $\mathcal{P}_{\mathsf{ex}}$ can be approximated in time polynomial both the precision and their representation. □

**Lemma 1.** *Every function in $\mathcal{P}_{\mathsf{ex}}$ is #·P-admissible.*

*Proof.* By Lemma 5, every function in $\mathcal{P}_{\mathsf{ex}}$ is P-computable. For Condition 2, note that the functions $\mathtt{uniform}(a, b)$ have a bounded domain, so that the condition is trivially satisfied. For the other functions, note that $\int_{-\infty}^{-2^{\delta(n)}} e^x \,\mathrm{d}x = e^{-2^{\delta(n)}}$. Since both functions are single exponential and bounded, it easily follows that these functions also satisfy Condition 2. We obtain that every function in $\mathcal{P}_{\mathsf{ex}}$ is #·P-admissible. □

## B.3 Complexity Upper Bounds

We first show a general lemma on products of probabilities with errors.

**Lemma 6.** *Let $N = \{p_1, \ldots, p_n\} \in 2^{[0,1]}$ and $m > n$. Then,*

- $\prod_{p_i \in N} (p_i + 2^{-m}) \leq \left(\prod_{p_i \in N} p_i\right) + 2^{-m+n-1}$, *and*

- $\prod_{p_i \in N} (p_i - 2^{-m}) \geq \left(\prod_{p_i \in N} p_i\right) - 2^{-m+n-1}$.

*Proof.* We only prove the first part of the lemma, as the second one can be proved analogously. The proof is by induction on the number of elements in $N$.

**Base Case** $N = \{p_1\}$. Then, $n = 1$ and $p_1 + 2^{-m} = p_1^{-m+1-1}$ directly follows.

**Inductive Step** Assume $N_n = \{p_1, \ldots, p_n\} \in 2^{[0,1]}$ and

$$\prod_{p_i \in N_n} (p_i + 2^{-m}) \leq \prod_{p_i \in N_n} p_i + 2^{-m+n-1}.$$

Let $p_{n+1} \in [0,1]$ and $N_{n+1} = N_n \cup \{p_{n+1}\}$. We establish the upper bound on the error for $N_{n+1}$.

$$\begin{aligned}
\prod_{p_i \in N_{n+1}} (p_i + 2^{-m}) &= \left(\prod_{p_i \in N_n} (p_i + 2^{-m})\right) \cdot (p_{n+1} + 2^{-m}) \\
&\leq \left(\left(\prod_{p_i \in N_n} p_i\right) + 2^{-m+n-1}\right) \cdot (p_{n+1} + 2^{-m}) \\
&= \left(\prod_{p_i \in N_{n+1}} p_i\right) + 2^{-m+n-1} \cdot p_{n+1} + \left(\prod_{p_i \in N_n} p_i\right) \cdot 2^{-m} + 2^{-2m+n-1}
\end{aligned}$$

Because $p_i \in [0,1]$, $\left(\prod_{p_i \in N_n} p_i\right) \in [0,1]$ and $2^{-m} \geq 2^{-2m-n-1}$, we can simplify

$$\begin{aligned}
\prod_{p_i \in N_{n+1}} (p_i + 2^{-m}) &\leq \left(\prod_{p_i \in N_{n+1}} p_i\right) + 2^{-m+n-1} + 2 \cdot 2^{-m} \\
&\leq \left(\prod_{p_i \in N_{n+1}} p_i\right) + 2^{-m+n-1} + 2^{-m+1}.
\end{aligned}$$

Furthermore, since $n > 1$, $2^{-m+n-1} \geq 2^{-m+1}$, and therefore

$$\begin{aligned}
\prod_{p_i \in N_{n+1}} (p_i + 2^{-m}) &\leq \left(\prod_{p_i \in N_{n+1}} p_i\right) + 2 \cdot 2^{-m+n-1} \\
&= \left(\prod_{p_i \in N_{n+1}} p_i\right) + 2^{-m+(n+1)-1}.
\end{aligned}$$

This concludes the proof. □

For a given input $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and $n$, we now follow the three steps for constructing the approximated measure space $M^a_{\mathcal{K},n}$, and show that each of the constructed measure spaces $M_1$, $M_2$ and $M^a_{\mathcal{K},n}$ satisfies the given error bounds on $P(\mathcal{K}, q)$ for any query $q$. In the following, for a query $q$ and $\star \in \{1, 2, a\}$, we use the notation $\mu^\star(q)$ as an abbreviation for $\mu^\star(\{w \in \Omega^\star_{\mathcal{A}} \mid (\mathcal{T}, w) \models q\})$, where $\mu^\star \in \{\mu_1, \mu_2, \mu^a_{\mathcal{K},n}\}$. Note that, while in the main text we only specify the set of possible worlds explicitly, we assume the event spaces $\Sigma_1$, $\Sigma_2$ and $\Sigma^a_{\mathcal{K},n}$ to be defined

analogously to how it is done in Section 3.1, and the measure functions $\mu_1$, $\mu_2$ and $\mu^a_{\mathcal{K},n}$ to be extended to be extended to measure functions that satisfy the conditions for measure spaces. Even though the approximated measure spaces are not probability measure spaces in the strict sense, for simplicity, we still speak of *probabilities* when speaking of values of the measure functions. Recall further that we assume w.l.o.g. all queries to only contain concepts $A$ that are concept names.

The first lemma is a direct result of Lemma 6 and the definition of #·P-feasible pdfs.

**Lemma 7** (Step 1)**.** *For every query $q$, we have*

$$|\mu_1(q) - \mu(q)| < 2^{-n-1}.$$

*Proof.* The maximal error produced in this step corresponds to the probability in $M_{\mathcal{A}}$ that any concrete feature value is outside the interval $[-2^{\max(\delta(n_v+n))}, 2^{\delta(n_v+n)})]$. For each concrete feature, this probability is maximally $2^{-n_v-n}$. Therefore, the probability of all concrete feature values to be inside $[-2^{\max(c,\delta(n_v+n))}, 2^{\max(c,\delta(n_v+n))})]$ is bounded by

$$\mu_1(\Omega_1) \geq (1 - 2^{-n_v-n})^{n_v}.$$

By Lemma 6, we thus have $\mu_1(\Omega_1) \geq 1 - 2^{-n-1}$. Because we only removed possible worlds, we have for any any query $q$ that $|\mu_1(q) - \mu(q)|$. □

**Lemma 8** (Step 2)**.** *For every query $q$, we have*

$$|\mu_2(q) - \mu_1(q)| < 2^{-n-2}.$$

*Proof.* Because $f(x) < 2^k$ for all $f \in \mathcal{P}$ and $x \in \mathbb{R}$, for every concrete feature $g(a)$ with $g(a) : f \in \mathbf{C}$ and every real number $x \in \mathbb{R}$, we have

$$\mu_{\mathcal{A}}(\{w \in \Omega_1 \mid w \models g(a) \in [x, x + 2^{-m}]\}) = \int_x^{x+2^{-m}} f(y)\,\mathrm{d}y < 2^{k-m}.$$

Therefore, for a real number $x \in [-2^{\delta(n_v+n)}, 2^{\delta(n_v+n)}]$ with $m$ bits after the binary point, we have $\mu_2(g(a) = x) < 2^{k+1-m}$. The probability in $M_2$ of any concrete feature value from $\mathbf{C}$ to have a value in occurring in a unary domain predicate in $\mathcal{K}$ is therefore bounded by $n_v n_c \cdot 2^{k+1-m}$. The probability of any two distinct concrete feature values from $\mathbf{C}$ to have the same value is bounded by $n_v^2 \cdot 2^{k+1-m}$. Therefore, we obtain as upper bound on the error for any query $q$:

$$|\mu_2(q) - \mu_1(q)| \leq n_v(n_v + n_c) \cdot 2^{k+1-m}.$$

By construction, we have $m > \log_2(n_v(n_v + n_c)) + n + k + 3$, and hence

$$|\mu_2(q) - \mu_1(q)| \leq 2^{-n-2}.$$ □

**Lemma 9** (Step 3)**.** *For every query $q$, we have*

$$|\mu^a_{\mathcal{K},n}(q) - \mu_2(q)| < 2^{-n-2}.$$

*Proof.* From the observations in the main text, we have for every $g(a) \sim f \in \mathbf{C}$ and $x, y \in \mathbb{R}$, $|\mu^a_{\mathcal{K},n}(g(a) \in [x, y]) - \mu_2(g(a) \in [x, y])| < 2^{-n_v-n-1}$. From Lemma 6, we therefore obtain $|\mu^a_{\mathcal{K},n}(\Omega_2) - \mu_2(\Omega_2)| < 2^{-n-2}$. Since this restricts the error also for every subset of $\Omega_2$, we obtain $|\mu^a_{\mathcal{K},n}(q) - \mu_2(q)| < 2^{-n-2}$ for every query $q$. □

Lemma 2 is now a direct consequence of Lemmata 7–9.

**Lemma 2.** *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a probabilistic KB, $q$ a query, $n \in \mathbb{N}$ and $M^a_{\mathcal{K},n}$ the $n$-approximated probability measure space for $\mathcal{K}$. Then,*

$$\mu^a_{\mathcal{K},n}(\{w \in \Omega^a_{\mathcal{K},n} \mid (\mathcal{T}, w) \models q\}) = \langle P_{\mathcal{K}}(q)\rangle_n.$$

It only remains to show that using the approximated measure spaces, we can define a counting problem corresponding to probabilistic query entailment that is in in $\#\cdot\mathcal{C}$, where $\mathcal{C}$ is the corresponding complexity of query entailment in the non-probabilistic case. As this corresponding problem will be used in the next sections to prove the lower bounds, we define it here formally:

**Definition 7.** The *counting variant of probabilistic query entailment* is the problem to compute the function $g$ that maps tuples $(\mathcal{K}, q, n)$ of a KB $\mathcal{K}$, a query $q$ and a precision $n$ in unary to a natural number such that

$$g(\mathcal{K}, q, n) = 2^{b_{\mathcal{K},n}} \cdot \langle P_{\mathcal{K}}(q)\rangle_n,$$

where $b_{\mathcal{K},n}$ is the maximal number of bits used in any probability in the $n$-approximated measure space $M^a_{\mathcal{K},n}$ for $\mathcal{K}$.

Note that $b_{\mathcal{K}}$ is polynomial in $|\mathcal{K}| + n$, so that Definition 7 fits the definition of a corresponding counting problem as defined in Section 5.

**Theorem 5.** *Let $\mathcal{L} \in \{\mathcal{EL}(\mathsf{R}_>), \mathcal{ALC}(\mathsf{R})\}$, $\mathcal{Q} \in \{AQ, UCQ\}$ and $\mathcal{C}$ be the combined/data complexity of $\mathcal{Q}$-entailment in classical $\mathcal{L}$ KBs. Then, the combined/data complexity of the counting variant of probabilistic $\mathcal{Q}$-entailment in probabilistic $\mathcal{L}$-knowledge bases is in $\#\cdot\mathcal{C}$.*

*Proof.* Let $\mathcal{L}$, $\mathcal{Q}$ and $\mathcal{C}$ be as in the theorem. Based on the approximated measure spaces $M^a_{\mathcal{K},n}$, we define a binary relation $R$ between tuples $(\mathcal{K}, q, n)$ of a KB $\mathcal{K}$, a query $q$ and a natural number $n$ in unary encoding, and tuples $(w, d)$ of possible worlds $w \in \Omega^a_{\mathcal{K},n}$ and natural numbers $d \in \mathbb{R}$ in binary encoding. Let $b_{\mathcal{K},n}$ be the (polynomial) number of bits required to represent the probability of a single possible world in $M^a_{\mathcal{K},n}$. The relation $R$ contains all tuples $((\mathcal{K}, q, n), (w, d))$ such that

1. $w \in W^a_{\mathcal{K},n}$,
2. $d < 2^{b_{\mathcal{K}}} \cdot \mu^a_{\mathcal{K},n}(\{w\})$, and
3. $w \models q$.

Conditions 1 and 2 can be checked in polynomial time, while Condition 3 can be checked in $\mathcal{C}$. Furthermore, each $|(w, d)|$ is polynomially bounded by $|\mathcal{K}| + n$. Hence, $R$ fulfils all requirements to show inclusion in $\#\cdot\mathcal{C}$, if we can show that it correctly captures probabilistic query entailment. Namely, we claim that

$$2^{b_{\mathcal{K},n}} \cdot \langle P(\mathcal{K}, q)\rangle_n = \|\{y \mid R((\mathcal{K}, q, n), y)\}\|.$$

Clearly, for any $w \in \Omega^a_{\mathcal{K},n}$, we have

$$2^{b_{\mathcal{K},n}} \cdot \mu^a_{\mathcal{A},n}(\{w\}) = \|\{d \mid R((\mathcal{K}, q, n), (w, d)\}\|.$$

$\mu^a_{\mathcal{K},n}(q)$ corresponds to the sum of all possible worlds $w \in \Omega^a_{\mathcal{K},n}$ s.t. $w \models q$, so that we obtain

$$2^{b_{\mathcal{K},n}} \cdot \mu^a_{\mathcal{K},n}(q) = \|\{y \mid R((\mathcal{K}, q, n), y)\}\|.$$

From this, our claim directly follows by Lemma 2. □

## B.4 Hardness of UCQ-Entailment in $\mathcal{EL}$ w.r.t Combined Complexity

We give a $\#\cdot$NP-lower bound under subtractive reductions for the combined complexity of the counting variant of probabilistic UCQ entailment in $\mathcal{EL}$ KBs. In fact, we show the stronger result on hardness for the counting variant of probabilistic *CQ* entailment in $\mathcal{EL}$ KBs. Turing reductions are generally too powerful to make meaningful hardness statements for a counting complexity class $\#\cdot\mathcal{C}$, if $\mathcal{C}$ is a complexity class larger P in the polynomial hierarchy. In fact, if $\mathcal{C}$ is in the polynomial hierarchy, every problem in $\#\cdot\mathcal{C}$ can be reduced to $\#\cdot$P under polynomial 1-Turing reductions [38]. Therefore, to make a meaningful statement on hardness for $\#\cdot$NP, we need a more restricted form of reduction that can distinguish between different levels of the polynomial counting hierarchy. For this reason, we use *subtractive reduction*.

Given a binary relation $R$, we denote by $R(x)$ the set $R(x) = \{y \mid R(x, y)\}$, and by $\#\cdot R$ the *counting problem associated with* $R$, which is to compute $\|R(x)\|$ given $x$. We can now define subtractive reductions after [16].

**Definition 8.** A counting problem $\#\cdot R$ reduces to another counting problem $\#\cdot S$ under *strong subtractive reductions* iff there are polynomially computable functions $f$ and $g$ such that for all words $x$

1. $S(g(x)) \subseteq S(f(x))$, and
2. $\|R(x)\| = \|S(f(x))\| - \|S(g(x))\|$.

Subtractive reduction is the transitive closure of strong subtractive reduction.

Subtractive reductions were introduced in [16] as a reduction method for the complexity classes $\#\cdot\Pi_n^p$, which are not closed under polynomial 1-Turing reductions, but are closed under subtractive reductions. The counting complexity classes $\#\cdot\Sigma_n^p$ (which includes $\#\cdot$NP) are not closed under subtractive reductions, in the sense that any problem in $\#\cdot\Pi_n^p$ can be reduced to $\#\cdot\Sigma_n^p$ under subtractive reductions. However, because we have $\#\cdot\Sigma_{n-1}^p \subseteq \#\cdot\Pi_{n-1}^p \subseteq \#\cdot\Sigma_n^p$ for all $n > 0$, subtractive reductions are restricted enough to differentiate different levels of the polynomial hierarchy. Specifically, since we already showed inclusion in $\#\cdot$NP for the counting variant of probabilistic UCQ entailment in $\mathcal{EL}$ KBs, and $\#\cdot$P is closed under subtractive reductions, subtractive reductions are sufficient for our purpose to show completeness in $\#\cdot$NP.

**Theorem 6.** *W.r.t. combined complexity, the counting variant of probabilistic CQ entailment in probabilistic $\mathcal{EL}$ KBs is $\#\cdot$NP-complete under subtractive reductions.*

*Proof.* $\#\cdot$NP-inclusion follows from Theorem 5, so that we only need to show hardness under subtractive reductions. The prototypical $\#\cdot$NP-complete problem is the following: given a propositional formula $\phi(x_0, \ldots, x_n)$ in CNF and a natural $m < n$, count the number of satisfying assignments of the QBF-formula $\exists x_m \ldots x_n : \phi(x_0, \ldots x_n)$ [16]. To show $\#\cdot$NP-hardness under subtractive reductions, we provide a strong subtractive reduction of this problem to probabilistic CQ entailment in $\mathcal{EL}$ KBs.

Let $\phi(x_0, \ldots, x_n)$ be a CNF-formula with clauses $c_1, \ldots, c_l$ and $m < n$. We construct a discrete probabilistic KB $\mathcal{K}_\phi$ and two queries $q_\phi^1$, $q_\phi^2$ such that the maximum number of bits after the binary point in any probability in the measure space $M_{\mathcal{K}_\phi}$ is $m$, the number of satisfying assignments of $\exists x_m \ldots x_n : \phi$ is

$$2^m \cdot \langle P_{\mathcal{K}_\phi}(q_\phi^1)\rangle_m - 2^m \cdot \langle P_{\mathcal{K}_\phi}(q_\phi^2)\rangle_m,$$

and furthermore, for every possible world $w \in \Omega_{\mathcal{K}_\phi}$, $w \models q_\phi^2$ implies $w \models q_\phi^1$. The latter property ensures that Condition 1 in Definition 8 is fulfilled for our reduction.

For every variable $x_i$ with $i \geq m$, we add the assertions $\mathsf{B}(x_i^+) : 1$, $\mathsf{B}(x_i^-) : 1$. For every variable $x_i$ with $i < m$, we add the assertions $\mathsf{B}^+(x_i^+) : 0.5$, $\mathsf{B}^-(x_i^-) : 0.5$. The TBox contains the axioms $\mathsf{B}^+ \sqsubseteq \mathsf{B}$ and $\mathsf{B}^- \sqsubseteq \mathsf{B}$. Intuitively, $\mathsf{B}$ marks which truth assignments to $x_i$ are available. For each clause $c_i \in \phi$ and each variable $x_j$ in $c_i$, we add $\mathsf{as}(c_i, x_j^+)$ if $x_j \in c_i$ and $\mathsf{as}(c_i, x_j^-)$ if $\neg x_j \in c_i$. Note that $m$ bits are sufficient to represent any probability in the corresponding measure space $M_{\mathcal{K}_\phi}$.

Let $q_c = \exists x_1, \ldots, x_n : B_i(x_i)$ be a query that is entailed in all possible worlds in which an assignment is available for each variable. Our queries $q_\phi^1$ and $q_\phi^2$ are now defined as follows:

$$q_\phi^1 = q_c \wedge \exists x_1, \ldots, x_l : \bigwedge_{1 \leq i \leq l} \mathsf{as}(c_i, x_i) \wedge \mathsf{B}(x_i)$$
$$q_\phi^2 = q_\phi^1 \wedge \exists x : \mathsf{B}^+(x) \wedge \mathsf{B}^-(x).$$

$q_\phi^1$ is satisfied in all possible worlds in which a satisfying truth-assignment is available, but also in those possible worlds in which more than one truth-assignment is available per free variable. Therefore, we define the query $q_\phi^2$ that is satisfied in those possible worlds which entail $q_\phi^1$ and in which for at least one free variable two truth values are available. One easily verifies that the inputs $(\mathcal{K}_\phi, q_\phi^1, n)$ and $(\mathcal{K}_\phi, q_\phi^2, n)$ can both be constructed from $\phi$ in polynomial time, that for every possible word $q$ in the associated measure space, $w \models q_\phi^2$ implies $w \models q_\phi^1$, and that the number of satisfying assignments of $\exists x_m \ldots x_n : \phi(x_0, \ldots, x_n)$ equals $2^m \cdot P_{\mathcal{K}_\phi}(q_\phi^1) - 2^m \cdot P_{\mathcal{K}_\phi}(q_\phi^2)$. We obtain that the counting variant of probabilistic CQ entailment in $\mathcal{EL}$ KBs is #·NP-hard under subtractive reductions w.r.t combined complexity . □

## B.5 Hardness of AQ-Entailment in $\mathcal{ALC}$ w.r.t. Data Complexity

While in the last section, we showed hardness under subtractive reductions, in order to distinguish #·NP and #·coNP, we need an even more restricted reduction, since every problem in #·coNP can be reduced to #·NP using subtractive reductions. We therefore use the most restricted reduction for counting problems, which is *parsimonious reduction* [38]. A parsimonious reduction from one counting problem to another is a polynomial reduction that preserves the number of solutions.

**Lemma 10.** *W.r.t. data complexity, the counting variant of probabilistic AQ-entailment in probabilistic $\mathcal{ALC}$ KBs is #·coNP-complete under parsimonious reductions.*

*Proof.* Since inclusion in #·coNP follows from Theorem 5, we only need to show #·coNP-hardness under parsimonious reductions. The prototypical #·coNP-complete problem is the following: given a 3DNF-formula $\phi(x_0, \ldots, x_n)$ and $m < n$, count the number of satisfying assignments of $\forall x_m \ldots x_n : \phi(x_0, \ldots, x_n)$. We define a probabilistic KB $\mathcal{K}_\phi = (\mathcal{T}_\forall, \mathcal{K}_\phi)$ such that the number of satisfying assignments of $\forall x_m \ldots x_n : \phi(x_0, \ldots, x_n)$ corresponds to $2^m \cdot \langle P_{\mathcal{K}_\phi}(A(f)) \rangle_m$.

We first define the Abox $\mathcal{A}_\phi$. For every variable $x_i$, we add the assertion $\mathsf{hasVar}(f, x_i)$. For every variable $x_i$ where $i \geq m$, we add the assertion $\mathsf{Q}(x_i) : 1$, while for every variable $x_i$ where $i < m$, we add the assertions $\mathsf{True}(x_i) : 0.5$ and $\mathsf{False}(x_i) : 0.5$.

To allow for a concise presentation of the following assertions, we define three mappings $r_i^l$, $i \in \{1, 2, 3\}$, that map a literal $l$ to a role based on their polarity: $r_i^x = \mathsf{p}_i$, $r_i^{\neg x} = \mathsf{n}_i$. $\mathcal{A}_\phi$ contains for each clause $c_i = l_1 \wedge l_2 \wedge l_3 \in \phi$ the following assertions:

$$\mathsf{hasClause}(f, c_i) : 1$$
$$r_1^{l_1}(c_i, \mathrm{var}(l_1)) : 1 \quad r_2^{l_2}(c_i, \mathrm{var}(l_2)) : 1 \quad r_3^{l_3}(c_i, \mathrm{var}(l_3)) : 1$$

Note that at most $m$ bits are required to represent any probability in $M_{\mathcal{A}_\phi}$. The TBox $\mathcal{T}_\forall$ contains the following axioms to define truth of 3DNF-clauses and -formulae.

$$\begin{aligned}
\mathsf{Q} &\sqsubseteq (\mathsf{True} \sqcup \mathsf{False}) \sqcap \neg(\mathsf{True} \sqcap \mathsf{False}) \\
\exists \mathsf{n}_1.\mathsf{False} \sqcap \exists \mathsf{n}_2.\mathsf{False} \sqcap \exists \mathsf{n}_3.\mathsf{False} &\sqsubseteq \mathsf{True} \\
\exists \mathsf{n}_1.\mathsf{False} \sqcap \exists \mathsf{n}_2.\mathsf{False} \sqcap \exists \mathsf{p}_3.\mathsf{True} &\sqsubseteq \mathsf{True} \\
&\vdots \\
\exists \mathsf{p}_1.\mathsf{True} \sqcap \exists \mathsf{p}_2.\mathsf{True} \sqcap \exists \mathsf{p}_3.\mathsf{True} &\sqsubseteq \mathsf{True} \\
\exists \mathsf{hasClause}.\mathsf{True} &\sqsubseteq \mathsf{True}
\end{aligned}$$

The concept $\mathsf{A}$ for the atomic query is defined as follows:

$$\mathsf{True} \sqcap \forall \mathsf{hasVar}.(\mathsf{True} \sqcup \mathsf{False}) \sqcap \neg \exists \mathsf{hasVar}.(\mathsf{True} \sqcap \mathsf{False}) \sqsubseteq A$$

The concept definition makes sure that $A(f)$ is entailed exactly in those possible worlds which (1) correspond to an assignment that makes the formula true (2) assigns every variable $x_i$ with $i < m$ at least one truth value, and (3) assigns every variable $x_i$ with $i < m$ at most one truth value. Correspondingly, the number of satisfying assignments of $\forall x_m \dots x_n : \phi(x_0, \dots, x_n)$ corresponds to $2^m \cdot \langle P_{\mathcal{K}_\phi}(A(f)) \rangle_m$. Note that the size of $\mathcal{A}_\phi$ is polynomial in the size of $\phi(x_0, \dots, x_n)$, while the size of $\mathcal{T}_\forall$ is independent of $\phi(x_0, \dots, x_n)$. We obtain that the data complexity of the counting problem corresponding to probabilistic AQ-entailment in $\mathcal{ALC}$ KBs is $\#\cdot$coNP-complete under parsimonious reductions. □