Most Probable Explanations for Probabilistic Database Queries

İsmail İlkan Ceylan and **Stefan Borgwardt**
Faculty of Computer Science
Technische Universität Dresden, Germany
ismail.ceylan@tu-dresden.de
stefan.borgwardt@tu-dresden.de

Thomas Lukasiewicz
Department of Computer Science
University of Oxford, UK
thomas.lukasiewicz@cs.ox.ac.uk

Abstract

Forming the foundations of large-scale knowledge bases, probabilistic databases have been widely studied in the literature. In particular, probabilistic query evaluation has been investigated intensively as a central inference mechanism. However, despite its power, query evaluation alone cannot extract all the relevant information encompassed in large-scale knowledge bases. To exploit this potential, we study two inference tasks; namely finding the *most probable database* and the *most probable hypothesis* for a given query. As natural counterparts of *most probable explanations (MPE)* and *maximum a posteriori hypotheses (MAP)* in probabilistic graphical models, they can be used in a variety of applications that involve *prediction* or *diagnosis* tasks. We investigate these problems relative to a variety of query languages, ranging from conjunctive queries to ontology-mediated queries, and provide a detailed complexity analysis.

1 Introduction

Research on building *large-scale probabilistic knowledge bases (PKBs)* has resulted in a number of systems including NELL [Mitchell *et al.*, 2015], Yago [Hoffart *et al.*, 2013], DeepDive [Shin *et al.*, 2015], and Google's Knowledge Vault [Dong *et al.*, 2014]. They have been used in a wide range of areas to automatically build structured knowledge bases. Most of them are based on the long tradition and foundations of *probabilistic databases* (PDBs) [Imieliski and Lipski, 1984; Suciu *et al.*, 2011], which define probability distributions over sets of classical databases.

Probabilistic query evaluation is the key inference task underpinning these systems. However, PKBs encompass rich, structured knowledge, for which alternative inference mechanisms beyond probabilistic query evaluation are needed. Inspired by the *maximal posterior probability* computations in Probabilistic Graphical Models (PGMs) [Pearl, 1988; Koller and Friedman, 2009], we investigate the problem of finding *most probable explanations* for probabilistic queries to exploit the potential of such large databases to their full extent.

Computing the *maximal posterior probability* of a distinguished set of variables, given evidence about another set of variables, is one of the key computational problems in PGMs. In its general form, this problem is studied under the name of *maximum a posteriori hypothesis*[1] *(MAP)*, where the hypothesis refers to an instantiation of a set of distinguished variables. The *most probable explanation (MPE)* is a special case of MAP, where the distinguished variables and the evidence variables cover all variables in the model. Maximal posterior probability computations have also been lifted to relational probabilistic models such as Markov Logic Networks (MLNs) [Richardson and Domingos, 2006].

Both MPE and MAP translate to probabilistic databases in a natural way through the rich structure of queries. The *most probable database* problem (analogous to MPE), first proposed in [Gribkoff *et al.*, 2014], is the problem of determining the (classical) database with the largest probability that satisfies a given query. Intuitively, the query defines constraints on the data, and the goal is to find the most probable database that satisfies these constraints. We also introduce a more intricate notion, called *most probable hypothesis*, which only asks for *partial* databases satisfying the query (analogous to MAP). The most probable hypothesis contains only tuples that contribute to the satisfaction condition of the query, which allows to more precisely pinpoint the most likely explanations of the query.

We study the computational complexity of the corresponding decision problems, denoted by MPD and MPH, respectively, for a variety of query languages. Our results provide detailed insights about the nature of these problems. We show that the data complexity of both problems is lower for *existential queries* than for *universal queries*. As expected, MPH usually has a higher complexity than MPD.

We extend our results towards *ontology-mediated queries (OMQs)*, which enrich the well-known class of *unions of conjunctive queries* with the power of ontological rules based on Datalog$^{\pm}$ (also called existential rules) [Baget *et al.*, 2011; Calì *et al.*, 2013]. This follows the tradition of *ontology-based data access* [Poggi *et al.*, 2008], which allows us to query PDBs in a more advanced manner. For OMQs, our analysis shows that the computational complexity of these problems can change significantly w.r.t. the ontology languages

[1]There exist other variants of these inference tasks [Kwisthout, 2011], and there are different naming conventions across communities; here, we use the terminology from the Bayesian Network (BN) literature [Darwiche, 2009].

under consideration and the complexity-theoretic assumptions employed. Our results provide tight complexity bounds for all Datalog$^\pm$ languages known in the literature. Detailed proofs of all results can be found at `https://lat.inf.tu-dresden.de/research/papers.html`.

2 Background and Motivation

We recall the basics of classical query languages and databases and briefly describe the *tuple-independent* PDB model.

2.1 Queries and Databases

We consider a relational vocabulary consisting of *finite* sets $\mathbf{R}$ of *predicates*, $\mathbf{C}$ of *constants*, and $\mathbf{V}$ of *variables*. A *term* is a constant or a variable. An *atom* is of the form $\mathtt{P}(s_1, \ldots, s_n)$, where $\mathtt{P}$ is an n-ary predicate, and $s_1, \ldots, s_n$ are terms. A *tuple* is an atom without variables.

A *conjunctive query (CQ)* is an existentially quantified formula $\exists \mathbf{x}\, \phi$, where ϕ is a conjunction of atoms, written as a comma-separated list. A *union of conjunctive queries (UCQ)* is a disjunction of CQs. If ϕ is an arbitrary Boolean combination of atoms, then we call $\exists \mathbf{x}\, \phi$ an *existential query ($\exists Q$)*, and $\forall \mathbf{x}\, \phi$ a *universal query ($\forall Q$)*. A *first-order query (FOQ)* is an arbitrary first-order formula. A query is *Boolean* if it has no free variables.

A *database* is a finite set of tuples. The central problem studied for databases is *query evaluation*: Finding all *answers* to a query Q over a database $\mathcal{D}$, which are assignments of the free variables in Q to constants such that the resulting first-order formula is satisfied in $\mathcal{D}$ in the usual sense. In the following, we consider only Boolean queries Q, and focus on the associated decision problem, i.e., deciding whether Q is satisfied in $\mathcal{D}$, denoted as usual by $\mathcal{D} \vDash \mathsf{Q}$.

2.2 Probabilistic Databases

The most elementary probabilistic database model is based on the tuple-independence assumption. We adopt this model and refer to [Suciu *et al.*, 2011] for details and alternatives. A probabilistic database induces a set of classical databases (called *worlds*), each associated with a probability value.

Formally, a *probabilistic database (PDB)* $\mathcal{P}$ is a finite set of *probabilistic tuples* of the form $\langle t : p \rangle$, where t is a tuple and $p \in [0,1]$, and, whenever $\langle t : p \rangle, \langle t : q \rangle \in \mathcal{P}$, then $p = q$. A PDB $\mathcal{P}$ assigns to every tuple t the probability p if $\langle t : p \rangle \in \mathcal{P}$, and otherwise the probability 0. Since we usually consider a single, fixed PDB, we denote this probability assignment simply by P. We concentrate on the well-known *possible worlds semantics*. Under the *tuple-independence assumption*, P induces the following *unique probability distribution* over classical databases $\mathcal{D}$:

$$\mathrm{P}(\mathcal{D}) \coloneqq \prod_{t \in \mathcal{D}} \mathrm{P}(t) \prod_{t \notin \mathcal{D}} (1 - \mathrm{P}(t)).$$

By the *worlds induced by* $\mathcal{P}$ we refer to those databases $\mathcal{D}$ with $\mathrm{P}(\mathcal{D}) > 0$. Query evaluation is also enriched by probabilistic information. More formally, the *probability* of a Boolean query Q w.r.t. P is $\mathrm{P}(\mathsf{Q}) \coloneqq \sum_{\mathcal{D} \vDash \mathsf{Q}} \mathrm{P}(\mathcal{D})$.

Example 1. Consider the PDB $\mathcal{P}_v$ given in Figure 1 and the conjunctive query

$$\mathsf{Q}_1 = \exists x, y\ \mathtt{Veg}(x) \land \mathtt{FriendOf}(x,y) \land \mathtt{Veg}(y),$$

Vegetarian	
alice	0.7
bob	0.9
chris	0.6

FriendOf		
alice	bob	0.7
alice	chris	0.8
bob	chris	0.1

Eats		
bob	spinach	0.7
chris	mussels	0.8
alice	broccoli	0.2

Meat	
shrimp	0.7
mussels	0.9
seahorse	0.3

Figure 1: The PDB $\mathcal{P}_v$, where each table represents a predicate and each row is a probabilistic tuple.

through which we can ask the probability of vegetarians being friends with vegetarians. In the given PDB, `alice`, `bob`, and `chris` are all vegetarians and friends with each other (with some probability). The query

$$\mathsf{Q}_1' = \exists y\ \mathtt{Veg}(\mathtt{bob}) \land \mathtt{FriendOf}(\mathtt{bob}, y) \land \mathtt{Veg}(y)$$

is a special case of Q_1 which asks whether `bob` has vegetarian friends. Its probability can be computed as $\mathrm{P}(\mathsf{Q}_1') = 0.9 \cdot 0.1 \cdot 0.6 = 0.054$. ■

This task is known as *probabilistic query evaluation*, and it has been studied extensively in PDBs. In a celebrated result by [Dalvi and Suciu, 2012], it has been shown that UCQs exhibit a data complexity dichotomy between P and #P for probabilistic query evaluation. However, while dealing with large-scale knowledge bases, alternative inference mechanisms are needed. Consider again our running example, and suppose that we have observed that Q_1' is true and would like to learn what best explains this observation w.r.t. the PDB $\mathcal{P}_v$. We revisit this example after providing a principled approach for dealing with such tasks.

3 Most Probable Explanations

We investigate the problem of finding *most probable explanations* for probabilistic database queries under two different semantics. Finding the *most probable database* is to determine the world with the largest probability that satisfies a given query, as formalized by [Gribkoff *et al.*, 2014]:

Definition 2. The *most probable database* for a query Q over a PDB $\mathcal{P}$ is

$$\arg\max_{\mathcal{D} \vDash \mathsf{Q}} \mathrm{P}(\mathcal{D}),$$

where $\mathcal{D}$ ranges over all worlds induced by $\mathcal{P}$.

Intuitively, a PDB defines a probability distribution over exponentially many classical databases, and the most probable database is the element in this collection that has the highest probability while still satisfying the query. This can be seen as the best instantiation of a probabilistic model, and hence analogous to MPE in BNs [Darwiche, 2009].

Example 3. Consider again the PDB $\mathcal{P}_v$ and the query

$$\mathsf{Q}_2 = \forall x, y\ \neg\mathtt{Veg}(x) \lor \neg\mathtt{Eats}(x,y) \lor \neg\mathtt{Meat}(y),$$

which defines the constraints of being a vegetarian, which is violated by the tuples `Veg(chris)`, `Eats(chris,mussels)` and `Meat(mussels)`. Hence, the most probable database for Q_2 cannot contain all three of them. It is easy to see that Veg(chris) is removed, as it has the lowest probability. Thus, the most probable database (in this case unique) contains all tuples of $\mathcal{P}_v$ that have a probability above 0.5, except for Veg(chris). Suppose that we have observed Q_1', and we are interested in finding an explanation for this observation under the constraint of Q_2, which is specified by $\mathsf{Q}_3 = \mathsf{Q}_1' \wedge \mathsf{Q}_2$. In this case, the most probable database must contain the tuples Veg(chris) and FriendOf(bob, chris), whereas Eats(chris, mussels) is omitted. ■

Finding the most probable database is important, as it identifies the most likely state of a PDB relative to a query. However, it has certain limitations, which are analogous to the limitations of MPE in PGMs [Koller and Friedman, 2009]. Most importantly, one is always forced to choose a *complete* database although the query usually affects only a subset of the tuples. In other words, it is usually not the case that the whole database is responsible for the goal query to be satisfied. To be able to more precisely pinpoint the explanations of a query, we introduce the following more intricate notion.

Definition 4. The *most probable hypothesis* for a query Q over a PDB $\mathcal{P}$ is

$$\arg\max_{\mathcal{H} \models \mathsf{Q}} \sum_{\mathcal{D} \models \mathcal{H}} \mathrm{P}(\mathcal{D}),$$

where $\mathcal{H}$ ranges over sets of tuples t and negated tuples $\neg t$ such that t occurs in $\mathcal{P}$, and $\models$ denotes the *open-world* entailment relation, i.e., $\mathcal{H} \models \mathsf{Q}$ holds iff all worlds induced by $\mathcal{P}$ that satisfy $\mathcal{H}$ also satisfy Q.

The sum inside the maximization evaluates to the product of the probabilities of the (negated) tuples in $\mathcal{H}$, and hence we denote it by $\mathrm{P}(\mathcal{H})$. Note that each $\mathcal{D} \models \mathcal{H}$ must satisfy Q, and thus the most probable database is a special case of the most probable hypothesis that has to contain all tuples from $\mathcal{P}$. In contrast, the most probable hypothesis contains tuples only if they contribute to the satisfaction of the query.

Example 5. The most probable hypothesis $\mathcal{H}$ for Q_3 consists of Veg(bob), Veg(chris), FriendOf(bob, chris), and ¬Eats(chris, mussels), which yields a probability of $\mathrm{P}(\mathcal{H}) = 0.9 \cdot 0.6 \cdot 0.1 \cdot (1 - 0.8) = 0.0108$. Since the most probable hypothesis contains less tuples, it is more informative than the most probable database for Q_3. ■

Whereas the most probable database represents full knowledge about all facts, which corresponds to the common closed-world assumption for (probabilistic) databases, the most probable hypothesis may leave tuples of $\mathcal{P}$ unresolved, which can be seen as a kind of open-world assumption (although the tuples that do not occur in $\mathcal{P}$ are still false).

Complexity Results

We formulate the decision problems MPD and MPH and investigate their computational complexity (see Table 1). We assume familiarity with computational complexity theory, and the distinction between *data* and *combined complexity*.

Definition 6. Let Q be a query, $\mathcal{P}$ a PDB, and $p \in (0, 1]$ a threshold. We denote by MPD (resp., MPH) the problem of deciding whether there exists a database $\mathcal{D}$ (resp., hypothesis $\mathcal{H}$) that satisfies Q with $\mathrm{P}(\mathcal{D}) \geq p$ (resp., $\mathrm{P}(\mathcal{H}) \geq p$).

Our first result concerns the well-known class of UCQs, and we observe that both MPD and MPH remain in *polynomial time* in the data complexity and in NP for the combined complexity.

Theorem 7. MPD *and* MPH *for UCQs can be decided in polynomial time in the data complexity and are* NP*-complete in the combined complexity.*

Intuitively, polynomial-time data complexity is ensured by the fact that UCQs are monotone queries: as a simple algorithm for MPD, consider all matches (of which there are polynomially many) of a given UCQ, and extend each match with all (negated) tuples from the PDB that have a probability greater than 0.5. This results in (polynomially many) classical databases, one of which is the most probable database. The polynomial data complexity result for MPH follows from similar arguments.

This result may seem immediate, as UCQs are monotone queries, but for MPD we can show an even stronger result that applies to all existential queries, even if negations in front of query atoms are allowed. For MPH, we show an additional hardness in the combined complexity.

Theorem 8. MPD *for* $\exists$*Q can be decided in polynomial time in the data complexity and is* NP*-complete in the combined complexity.* MPH *for* $\exists$*Q can be decided in* Σ_2^{P} *in the data complexity and is* Σ_3^{P}*-complete in the combined complexity.*

Determining the precise data complexity of MPH for $\exists$Q is left as an open problem.

For universally quantified queries, MPD becomes harder even in the data complexity. Given the rich structure of the query, it becomes possible to encode non-deterministic choices over all possible databases. It is important to note that a similar hardness result was also obtained in [Gribkoff *et al.*, 2014] (although on a different query).

Theorem 9. MPD *for* $\forall$*Q is* NP*-complete in the data complexity.*

We observe a similar phenomenon for MPH, where the increase in the complexity is even more dramatic.

Theorem 10. MPH *for* $\forall$*Q is* Σ_2^{P}*-complete in the data complexity.*

Note that also MPE and MAP differ in terms of computational complexity: in BNs, the former is NP-complete [Shimony, 1994; Littman, 1999] and the latter $\mathrm{NP}^{\mathrm{PP}}$-complete [Park and Darwiche, 2004a]. Differently, MPH remains in $\Sigma_2^{\mathrm{P}} \subseteq \mathrm{NP}^{\mathrm{PP}}$, since computing the probability of the hypothesis can be done in polynomial time due to the tuple-independence assumption. As we will examine later, allowing ontological rules to induce correlations on the tuples results in different complexities, which are more comparable to PGMs, which can express conditional dependencies between their variables. Our final result for this section concerns the combined complexity of both problems for $\forall$Q.

Theorem 11. MPD *and* MPH *for* $\forall$*Q are* Σ_2^{P}*-complete in the combined complexity.*

If we consider arbitrary FOQs, it is easy to show that the data complexity remains the same as for $\forall$Qs, and the combined complexity is still PSPACE, as for classical databases.

Moreover, all our combined complexity results for PDBs hold even in the case where the arity of the predicates is bounded by some constant, commonly known as the *bounded-arity (ba) combined complexity* (see Table 1). The reason is that our hardness results use only predicates of a bounded arity.

4 Most Probable Explanations for OMQs

We now study the problems in the presence of ontological rules, which add deductive power to PDBs. The benefits of using of ontologies for large-scale PKB completion are well known [Jung and Lutz, 2012; Borgwardt *et al.*, 2017]. From a broader perspective, extending tuple-independent PDBs with logical rules is an old idea, aiming to induce correlations on a logical level, and thus to relax the tuple independence, resulting in very powerful formalisms [Poole, 1993; 1997].

In the remainder of this paper, we restrict ourselves to UCQs, but in exchange consider additional knowledge encoded through an ontology. In the simplest case, we can formulate *negative constraints (NCs)* like

$$\forall x, y\ \mathtt{Veg}(x) \wedge \mathtt{Eat}(x, y) \wedge \mathtt{Meat}(y) \to \bot,$$

which imposes the same constraints as Q_2. NCs are a special case of denial constraints over databases [Staworko and Chomicki, 2010]. We also allow to formulate more general ontological knowledge in the form of *tuple-generating dependencies (TGDs)*. For instance, the first-order formulas

$$\forall x, y\ \mathtt{FriendOf}(x, y) \to \mathtt{FriendOf}(y, x),$$
$$\forall x\ \mathtt{Veg}(x) \to \exists y\ \mathtt{Knows}(x, y) \wedge \mathtt{Veg}(y)$$

are TGDs stating that the friend relation is symmetric and that every vegetarian knows another vegetarian. In particular, TGDs can express the well-known inclusion dependencies and join dependencies from database theory. Taking all parts together, we are talking about *ontology-mediated queries*, which are UCQs in combination with a so-called Datalog$^\pm$ ontology, i.e., a finite set of NCs and TGDs [Calì *et al.*, 2012]. We will pay particular attention to the case where only NCs are allowed, as this is closest to the constraints over PDBs that we described in the previous examples, and it gives us a baseline for the computational complexity.

More formally, an *NC* ν is an FO formula $\forall \mathbf{x}\, \varphi(\mathbf{x}) \to \bot$, where $\varphi(\mathbf{x})$ is a conjunction of atoms, called the *body* of ν, and $\bot$ is the truth constant *false*. A *TGD* σ is an FO formula $\forall \mathbf{x}\, \varphi(\mathbf{x}) \to \exists \mathbf{y}\, \mathtt{P}(\mathbf{x}, \mathbf{y})$, where $\varphi(\mathbf{x})$ is a conjunction of atoms, called the *body* of σ, and $\mathtt{P}(\mathbf{x}, \mathbf{y})$ is an atom, called the *head* of σ. A *(Datalog$^\pm$) program* (or *ontology*) Σ is a finite set of NCs and TGDs.[2] An *ontology-mediated query (OMQ)* (Q, Σ) consists of a program Σ and a Boolean UCQ Q.

For the semantics, we extend the vocabulary by an infinite set $\mathbf{N}$ of *nulls*. An instance I is a possibly infinite set of tuples that may additionally contain nulls; it *satisfies* a TGD or NC σ if $I \vDash \sigma$, where $\vDash$ denotes standard FO entailment. I satisfies a program Σ, written $I \vDash \Sigma$, if I satisfies each formula in Σ. The set $mods(\mathcal{D}, \Sigma)$ of *models* of a database $\mathcal{D}$ relative to a program Σ is $\{I \mid I \supseteq \mathcal{D} \text{ and } I \vDash \Sigma\}$. A database $\mathcal{D}$ is *consistent* w.r.t. Σ if $mods(\mathcal{D}, \Sigma)$ is non-empty. The OMQ (Q, Σ) is *entailed* by $\mathcal{D}$, denoted $\mathcal{D} \vDash (\mathsf{Q}, \Sigma)$, if $I \vDash \mathsf{Q}$ holds for all $I \in mods(\mathcal{D}, \Sigma)$. Observe that consistency of $\mathcal{D}$ w.r.t. Σ can thus be written as $\mathcal{D} \nvDash (\bot, \Sigma)$, or equivalently, $\mathcal{D} \nvDash (\mathsf{Q}_\bot, \Sigma^+)$, where $\mathsf{Q}_\bot$ is the UCQ obtained from the disjunction of the bodies of all NCs in Σ, and Σ^+ contains all TGDs of Σ. The *probability* of an OMQ over a PDB is defined as usual, by summing up over the consistent worlds that entail the query.

[2]We omit the universal quantifiers in TGDs and NCs, and use commas (instead of $\wedge$) for conjoining atoms for ease of presentation.

In general, the entailment problem is undecidable [Beeri and Vardi, 1981], which motivated syntactic fragments of TGDs (but not the NCs) [Calì *et al.*, 2012; Baget *et al.*, 2011; Krötzsch and Rudolph, 2011; Calì *et al.*, 2013; Fagin *et al.*, 2005]. We focus on only a few of these classes here (see Table 1). If there are no TGDs at all, i.e., $\Sigma^+ = \varnothing$, then we denote the query language by $\mathsf{OMQ}_{\mathsf{NC}}$. One of the most important classes, *guarded* programs ($\mathsf{OMQ}_{\mathsf{G}}$) allow only such TGDs that have a body atom (the *guard*) that contains all body variables. In the special case of *linear* programs ($\mathsf{OMQ}_{\mathsf{L}}$), the body may consist of only one atom. In *frontier-guarded* programs ($\mathsf{OMQ}_{\mathsf{FG}}$), only the frontier variables, i.e., those shared by the body and the head, need to be guarded. We pay particular attention to the class of *guarded and full* programs ($\mathsf{OMQ}_{\mathsf{GF}}$), which does not allow existentially quantified variables, and is one of the least expressive Datalog$^\pm$-based query languages with a P-complete data complexity. At the upper end of the expressivity spectrum, we observe classes like *weakly guarded* ($\mathsf{OMQ}_{\mathsf{WG}}$) programs, whose data complexity is already EXP-complete. From a theoretical perspective, the class of *acyclic* programs ($\mathsf{OMQ}_{\mathsf{A}}$), which do not allow cyclic dependencies between predicates, is of interest, because it has a non-deterministic combined complexity, which leads to a different behavior.

A key paradigm in OMQ answering is the *FO-rewritability* of queries: an OMQ (Q, Σ) is *FO-rewritable* if there exists a Boolean UCQ Q_Σ such that, for all consistent databases $\mathcal{D}$, it holds that $\mathcal{D} \vDash (\mathsf{Q}, \Sigma)$ iff $\mathcal{D} \vDash \mathsf{Q}_\Sigma$. Intuitively, the rewritten query Q_Σ can be answered directly over the database, without referring to the Datalog$^\pm$ program. A class $\mathsf{OMQ}_{\mathsf{X}}$ is *FO-rewritable* if all its OMQs are FO-rewritable; in this case, the OMQ entailment problem for $\mathsf{OMQ}_{\mathsf{X}}$ has a data complexity of AC^0. Of the classes mentioned above, only $\mathsf{OMQ}_{\mathsf{NC}}$, $\mathsf{OMQ}_{\mathsf{L}}$, and $\mathsf{OMQ}_{\mathsf{A}}$ have this property.

In addition to data complexity, *ba*-combined complexity, and combined complexity, for OMQs we also consider the *fixed-program (fp) combined complexity*, where the Datalog$^\pm$ program is viewed as fixed, but both (P)DB and query are considered to be part of the input. The *ba*-combined complexity is of interest for Description Logics [Baader *et al.*, 2003], some of which can be considered to be Datalog$^\pm$ languages with arity at most 2.

4.1 Most Probable Databases for OMQs

For ontology-mediated queries, models are restricted to those that are consistent with the ontology. In other words, the constraints are considered separately from the (existential) query; thus, the definitions of MPD and MPH have to be adapted accordingly. The main difference is that we now consider only the *consistent* worlds induced by the PDB, and thus maximize only over consistent worlds.

Query Languages	**Most Probable Database**				**Most Probable Hypothesis**			
	data	fp**-comb.**	ba**-comb.**	**comb.**	**data**	fp**-comb.**	ba**-comb.**	**comb.**
UCQ	in P	—	NP	NP	in P	—	NP	NP
$\exists$Q	in P	—	NP	NP	in Σ_2^{P}	—	Σ_3^{P}	Σ_3^{P}
$\forall$Q	NP	—	Σ_2^{P}	Σ_2^{P}	Σ_2^{P}	—	Σ_2^{P}	Σ_2^{P}
FOQ	NP	—	PSPACE	PSPACE	Σ_2^{P}	—	PSPACE	PSPACE
OMQ$_{\mathsf{NC}}$	NP	NP	Σ_2^{P}	Σ_2^{P}	PP	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$
OMQ$_{\mathsf{L}}$	NP	NP	Σ_2^{P}	PSPACE	PP	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$	PSPACE
OMQ$_{\mathsf{A}}$	NP	NP	P$^{\mathrm{NE}}$	P$^{\mathrm{NE}}$	PP	NP$^{\mathrm{PP}}$	P$^{\mathrm{NE}}$	P$^{\mathrm{NE}}$
OMQ$_{\mathsf{GF}}$	NP	NP	Σ_2^{P}	EXP	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$	EXP
OMQ$_{\mathsf{G}}$	NP	NP	EXP	2EXP	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$	EXP	2EXP
OMQ$_{\mathsf{FG}}$	NP	NP	2EXP	2EXP	NP$^{\mathrm{PP}}$	NP$^{\mathrm{PP}}$	2EXP	2EXP
OMQ$_{\mathsf{WG}}$	EXP	EXP	EXP	2EXP	EXP	EXP	EXP	2EXP

Table 1: Complexity results for MPD and MPH for a wide range of queries: all listed results are original contributions of this paper except the data complexity for $\forall$Q. This latter result has been strengthened towards the weaker representation OMQ$_{\mathsf{NC}}$.

Definition 12. The *most probable database* for an OMQ (Q, Σ) over a PDB $\mathcal{P}$ is

$$\underset{\mathcal{D}\models(\mathsf{Q},\Sigma),\, mods(\mathcal{D},\Sigma)\neq\varnothing}{\arg\max} \quad \mathrm{P}(\mathcal{D}).$$

As before, we consider the decision problem MPD, which checks for the existence of a world $\mathcal{D}$ as above that exceeds a given probability threshold p. Our complexity results for this problem are summarized in the lower left part of Table 1.

A naive approach to solve MPD is to guess the database $\mathcal{D}$, and then check that it entails the given OMQ, does *not* entail the query $(\mathsf{Q}_\perp, \Sigma^+)$, and exceeds the probability threshold. Since the probability can be computed in polynomial time, the problem can be decided by an NP Turing machine using an oracle to check OMQ entailment.

Theorem 13. *If entailment for OMQ*$_{\mathsf{X}}$ *is in a complexity class* $\mathbf{C}$*, then* MPD *for OMQ*$_{\mathsf{X}}$ *is in* NP$^{\mathbf{C}}$ *(under the same complexity assumptions).*

By a reduction from 3-colorability, we show that MPD is NP-hard already in the data complexity, even if we only use NCs, i.e., the query and the positive program Σ^+ are empty. This strengthens our previous result about $\forall$Qs, since NCs can be expressed by universal queries, but are not allowed to use negated atoms.

Theorem 14. MPD *for OMQ*$_{\mathsf{NC}}$ *is* NP*-hard in the data complexity, even for* $\mathsf{Q} = \top$.

We show a matching upper bound even in the fp-combined complexity, by reconsidering the approach from Theorem 13. Since here the query $(\mathsf{Q}_\perp, \Sigma^+)$ is fixed, for the non-entailment check, we can refer to the *data complexity* of OMQ entailment. Assuming that this is possible in P, and further that the fp-combined complexity does not exceed NP, then the whole test can be done by a single non-deterministic Turing machine in polynomial time. This insight yields an upper bound of NP for most of the classes we consider.

Theorem 15. *If entailment for OMQ*$_{\mathsf{X}}$ *is in* NP *in the fp-combined complexity and in* P *in the data complexity, then* MPD *for OMQ*$_{\mathsf{X}}$ *is in* NP *in the fp-combined complexity.*

Under ba-combined complexity assumptions, we observe an increase in complexity, which intuitively comes from the fact that the query $(\mathsf{Q}_\perp, \Sigma^+)$ is not fixed anymore.

Theorem 16. MPD *for OMQ*$_{\mathsf{NC}}$ *is* Σ_2^{P}*-hard in ba-combined complexity, even for* $\mathsf{Q} = \top$.

We obtain a matching upper bound from Theorem 13 if $\mathbf{C}$ = NP. Most of the remaining hardness results follow from the complexity of classical OMQ entailment, since an OMQ (Q, Σ) is entailed by a consistent database $\mathcal{D}$ iff the most probable database w.r.t. $\{\langle t : 1\rangle \mid t \in \mathcal{D}\}$ has probability 1. In combination with Theorem 13, this yields tight complexity results for large deterministic classes like PSPACE or EXP. This leaves open only the case of OMQ$_{\mathsf{A}}$, for which entailment is NEXP-complete in the (ba-)combined complexity, which yields an upper bound of NP$^{\mathrm{NExp}}$ = P$^{\mathrm{NExp}}$ = P$^{\mathrm{NE}}$ due to Theorem 13 and results from [Hemachandra, 1989]. We show that this bound is tight, using a reduction from a P$^{\mathrm{NExp}}$-complete version of the *tiling problem* [Fürer, 1983].

Theorem 17. MPH *for OMQ*$_{\mathsf{A}}$ *is* P$^{\mathrm{NE}}$*-hard in the ba-combined complexity.*

4.2 Most Probable Hypotheses for OMQs

Similarly to MPD, we have to update the definition of MPH to take into account only consistent worlds.

Definition 18. The *most probable hypothesis* for an OMQ (Q, Σ) over a PDB $\mathcal{P}$ is

$$\underset{\mathcal{H}\models(\mathsf{Q},\Sigma)}{\arg\max} \sum_{\substack{\mathcal{D}\supseteq\mathcal{H} \\ mods(\mathcal{D},\Sigma)\neq\varnothing}} \mathrm{P}(\mathcal{D}),$$

where $\mathcal{H}$ is a set of (non-probabilistic) tuples t occurring in $\mathcal{P}$.

Since we consider only monotone queries (i.e., UCQs), we do not have to include negated tuples in the hypothesis.

To solve MPH, one can guess a hypothesis, and then check whether it entails the query, and whether the probability mass of its consistent extensions exceeds the given threshold. The latter part can be done by a PP Turing machine with an oracle for OMQ entailment (by normalizing the probability of the worlds such that the threshold is exactly 0.5). The oracle can be used also for the initial entailment check.

Theorem 19. *If entailment for OMQ*$_{\mathsf{X}}$ *is in a complexity class* $\mathbf{C}$*, then* MPH *for OMQ*$_{\mathsf{X}}$ *is in* $\mathrm{NP}^{\mathrm{PP}^{\mathbf{C}}}$ *(under the same complexity assumptions).*

For any $\mathbf{C} \subseteq \mathrm{PH}$, this yields $\mathrm{NP}^{\mathrm{PP}^{\mathrm{PH}}} = \mathrm{NP}^{\mathrm{P}^{\mathrm{PP}}} = \mathrm{NP}^{\mathrm{PP}}$ as an upper bound, due to a result from [Toda, 1989]. Except for PP, all other upper bounds in the lower right part of Table 1 also follow from this observation. For $\mathbf{C}$ = NEXP, the whole $\mathrm{PP}^{\mathrm{NExp}}$ computation can be done by an NEXP oracle, and hence we again obtain $\mathrm{NP}^{\mathrm{NExp}} = \mathrm{P}^{\mathrm{NE}}$ in this case.

On the other hand, we can transfer most of the lower bounds from MPD by a simple reduction.

Theorem 20. *If* MPD *for OMQ*$_{\mathsf{X}}$ *is hard for a complexity class* $\mathbf{C}$*, then* MPH *for OMQ*$_{\mathsf{X}}$ *is also hard for* $\mathbf{C}$ *(under any complexity measure except for the data complexity).*

We now discuss the more interesting cases below PSPACE. For FO-rewritable classes of OMQs, we obtain a data complexity of only PP. This is due to the fact that TGDs can be compiled into the query, and the observation of Theorem 8 that existential queries can be processed using polynomially many steps. Nevertheless, in each step, we have to compute a non-trivial sum over the *consistent* worlds, which can be done in PP. Finally, the polynomially many PP checks can be compiled into a single PP operation [Beigel *et al.*, 1995].

Theorem 21. *If entailment for OMQ*$_{\mathsf{X}}$ *is in* AC^0 *in the data complexity, then* MPH *for OMQ*$_{\mathsf{X}}$ *is* PP*-complete in the data complexity.*

To frame this result, we show $\mathrm{NP}^{\mathrm{PP}}$-hardness both for OMQ$_{\mathsf{GF}}$ (the prototypical non-FO-rewritable class) in the data complexity, and for OMQ$_{\mathsf{NC}}$ in the fp-combined complexity. We use two similar reductions from a problem in the polynomial-time counting hierarchy [Wagner, 1986]. In the first one, we rely on the power of guarded and full TGDs; in the second, we use a non-fixed query and a careful choice of probabilities to simulate the effect of these TGDs.

Theorem 22. MPH *is* $\mathrm{NP}^{\mathrm{PP}}$*-hard for OMQ*$_{\mathsf{GF}}$ *in the data complexity, and for OMQ*$_{\mathsf{NC}}$ *in the fp-combined complexity.*

We also observe a dichotomy in the data complexity of MPH, which follows from the dichotomy for probabilistic query evaluation [Dalvi and Suciu, 2012]. Note that the PP-hardness holds only under Turing reductions.

Lemma 23 (Dichotomy)**.** *For FO-rewritable classes OMQ*$_{\mathsf{X}}$*,* MPH *is either in* P *or* PP*-hard in the data complexity.*

Our results apply to all decidable Datalog$^{\pm}$ languages from the literature, due to the generic nature of our theorems and the known complexity results for OMQ entailment [Calì *et al.*, 2012; Baget *et al.*, 2011; Krötzsch and Rudolph, 2011; Calì *et al.*, 2013]. For example, *full* sets of TGDs (F) behave like OMQ$_{\mathsf{GF}}$, and *linear full* (LF) and *acyclic full* (AF) sets of TGDs exhibit the same complexities as OMQ$_{\mathsf{L}}$.

5 Related Work

Our work is inspired by the *maximal posterior probability* computations in PGMs [Pearl, 1988; Koller and Friedman, 2009]. It is well-known that they are a form of *abduction*, which clearly also applies to the problems studied here. Maximal posterior probability computations are central in PGMs and in statistical relational learning. However, analogous problems have not been studied in depth in the context of PDBs. To our knowledge, the only work in this direction is on most probable databases [Gribkoff *et al.*, 2014], which we use as a starting point.

Probabilistic query evaluation has been investigated in depth in the literature [Imieliski and Lipski, 1984; Suciu *et al.*, 2011]. In fact, it is known to be either in P or #P-hard for UCQs in the data complexity [Dalvi and Suciu, 2012]. Extensions of PDBs with ontological rules, in particular with Datalog$^{\pm}$, are also well-covered in the literature [Gottlob *et al.*, 2013; Borgwardt *et al.*, 2017]. The focus of these works is on probabilistic query answering.

Most of our data complexity results are covered by the classes NP, Σ_2^{P}, PP, and $\mathrm{NP}^{\mathrm{PP}}$. Though intractable, they are at the core of many important problems, which motivated a body of work tailored towards scalable algorithms for these classes. There is an immediate connection between MPE and weighted MAX-SAT [Sang *et al.*, 2007]. Similarly, advances in knowledge compilation [Park and Darwiche, 2004b; Pipatsrisawat and Darwiche, 2009] and approximate model counting [Chakraborty *et al.*, 2016; Fremont *et al.*, 2017] are tailored to achieve optimal, scalable algorithms for problems in classes such as PP and $\mathrm{NP}^{\mathrm{PP}}$. Both MPD and MPH can be cast into their propositional variants using the lineage representation of queries, which gives immediate access to such algorithms. However, there is need for future work in this direction, as grounding is not an optimal way of handling these problems; performing inference directly on FO-structures is known to be more efficient.

6 Summary and Outlook

We studied two inference tasks for PDBs; namely finding the most probable database and the most probable hypothesis for a given query. The focus of this paper was to determine the precise complexity of these problems, and we provided a detailed analysis for these problems relative to a variety of query languages, ranging from conjunctive queries to ontology-mediated queries. The main focus of future work is on the one hand to extend these results to other classes (such as equality generating dependencies) and on the other hand to obtain even more refined results (such as classification results).

Acknowledgments

This work was supported by DFG in the CRC 912 (HAEC), GRK 1907 (RoSI), and the project BA 1122/19-1 (GoAsQ), and by the UK EPSRC grants EP/J008346/1, EP/L012138/1, EP/M025268/1, and by The Alan Turing Institute under the EPSRC grant EP/N510129/1.

References

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[Baget *et al.*, 2011] Jean-François Baget, Marie-Laure Mugnier, Sebastian Rudolph, and Michaël Thomazo. Walking the complexity lines for generalized guarded existential rules. In *Proc. IJCAI*, 2011.

[Beeri and Vardi, 1981] Catriel Beeri and Moshe Y. Vardi. The implication problem for data dependencies. In *Proc. ICALP*, 1981.

[Beigel *et al.*, 1995] Richard Beigel, Nick Reingold, and Daniel Spielman. PP is closed under intersection. *JCSS*, 50(2):191–202, 1995.

[Borgwardt *et al.*, 2017] Stefan Borgwardt, İsmail İlkan Ceylan, and Thomas Lukasiewicz. Ontology-mediated queries for probabilistic databases. In *Proc. AAAI*, 2017.

[Calì *et al.*, 2012] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *JWS*, 14:57–83, 2012.

[Calì *et al.*, 2013] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *JAIR*, 48:115–174, 2013.

[Chakraborty *et al.*, 2016] Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic SAT calls. In *Proc. IJCAI*, 2016.

[Dalvi and Suciu, 2012] Nilesh Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6):1–87, 2012.

[Darwiche, 2009] Adnan Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[Dong *et al.*, 2014] Xin Luna Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Patrick Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge Vault: A web-scale approach to probabilistic knowledge fusion. In *Proc. SIGKDD*, 2014.

[Fagin *et al.*, 2005] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. *TCS*, 336(1):89–124, 2005.

[Fremont *et al.*, 2017] Daniel Fremont, Markus Rabe, and Sanjit Seshia. Maximum model counting. In *Proc. AAAI*, 2017.

[Fürer, 1983] Martin Fürer. The computational complexity of the unconstrained limited domino problem (with implications for logical decision problems). In *Logic and Machines*, 1983.

[Gottlob *et al.*, 2013] Georg Gottlob, Thomas Lukasiewicz, Maria Vanina Martinez, and Gerardo I. Simari. Query answering under probabilistic uncertainty in Datalog+/– ontologies. *AMAI*, 69(1):37–72, 2013.

[Gribkoff *et al.*, 2014] Eric Gribkoff, Guy Van den Broeck, and Dan Suciu. The most probable database problem. In *Proc. BUDA*, 2014.

[Hemachandra, 1989] Lane A. Hemachandra. The strong exponential hierarchy collapses. *JCSS*, 39(3):299–322, 1989.

[Hoffart *et al.*, 2013] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *AIJ*, 194:28–61, 2013.

[Imieliski and Lipski, 1984] Tomasz Imieliski and Witold Lipski. Incomplete information in relational databases. *J. ACM*, 31(4):761–791, 1984.

[Jung and Lutz, 2012] Jean Christoph Jung and Carsten Lutz. Ontology-based access to probabilistic data with OWL QL. In *Proc. ISWC*, 2012.

[Koller and Friedman, 2009] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[Krötzsch and Rudolph, 2011] Markus Krötzsch and Sebastian Rudolph. Extending decidable existential rules by joining acyclicity and guardedness. In *Proc. IJCAI*, 2011.

[Kwisthout, 2011] Johan Kwisthout. Most probable explanations in Bayesian networks: Complexity and tractability. *IJAR*, 52(9):1452–1469, 2011.

[Littman, 1999] Michael L. Littman. Initial experiments in stochastic satisfiability. In *Proc. AAAI*, 1999.

[Mitchell *et al.*, 2015] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapa Nakashole, Emmanouil Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard Wang, Derry Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. Never-ending learning. In *Proc. AAAI*, 2015.

[Park and Darwiche, 2004a] James D. Park and Adnan Darwiche. Complexity results and approximation strategies for MAP explanations. *JAIR*, 21(1):101–133, 2004.

[Park and Darwiche, 2004b] James D. Park and Adnan Darwiche. A differential semantics for jointree algorithms. *AIJ*, 156(2):197–216, 2004.

[Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[Pipatsrisawat and Darwiche, 2009] Knot Pipatsrisawat and Adnan Darwiche. A new d-DNNF-based bound computation algorithm for functional E-MAJSAT. In *Proc. IJCAI*, 2009.

[Poggi *et al.*, 2008] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. *JDS*, 10:133–173, 2008.

[Poole, 1993] David Poole. Probabilistic Horn abduction and Bayesian networks. *AIJ*, 64(1):81–129, 1993.

[Poole, 1997] David Poole. The independent choice logic for modelling multiple agents under uncertainty. *AIJ*, 94(1):7–56, 1997.

[Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov logic networks. *ML*, 62(1):107–136, 2006.

[Sang *et al.*, 2007] Tian Sang, Paul Beame, and Henry Kautz. A dynamic approach to MPE and weighted MAX-SAT. In *Proc. IJCAI*, 2007.

[Shimony, 1994] Eyal Solomon Shimony. Finding MAPs for belief networks is NP-hard. *AIJ*, 68(2):399–410, 1994.

[Shin *et al.*, 2015] Jaeho Shin, Sen Wu, Feiran Wang, Christopher De Sa, Ce Zhang, and Christopher Ré. Incremental knowledge base construction using DeepDive. In *Proc. VLDB*, 2015.

[Staworko and Chomicki, 2010] Sławomir Staworko and Jan Chomicki. Consistent query answers in the presence of universal constraints. *Inf. Syst.*, 35(1):1–22, 2010.

[Suciu *et al.*, 2011] Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. *Probabilistic Databases*. Morgan & Claypool, 2011.

[Toda, 1989] Seinosuke Toda. On the computational power of PP and ⊕P. In *Proc. SFCS*, 1989.

[Wagner, 1986] Klaus W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Inf.*, 23(3):325–356, 1986.