What Does a Query Answer Tell You? Informativeness of Query Answers for Knowledge Bases

Luca Andolfi, Gianluca Cima, Marco Console, Maurizio Lenzerini

Sapienza University of Rome
{andolfi, cima, console, lenzerini}@diag.uniroma1.it

Abstract

Query answering for Knowledge Bases (KBs) amounts to extracting information from the various models of a KB, and presenting the user with an object that represents such information. In the vast majority of cases, this object consists of those tuples of constants that satisfy the query expression either in every model (*certain answers*) or in some model (*possible answers*). However, similarly to the case of incomplete databases, both these forms of answers are a lossy representation of all the knowledge inferable from the query and the queried KB. In this paper, we illustrate a formal framework to characterize the information that query answers for KBs are able to represent. As a first application of the framework, we study the informativeness of current query answering approaches, including the recently introduced partial answers. We then define a novel notion of answers, allowing repetition of variables across answer tuples. We show that these answers are capable of representing a meaningful form of information, and we also study their data complexity properties.

1 Introduction

A Knowledge Base (KB) is a symbolic representation of the knowledge pertaining to a specific domain of interest. In the logical approach to KBs (Levesque and Lakemeyer 2001), this representation consists of a *logical theory*, i.e., a finite set of assertions expressed in some formal language. The *models* of this theory define the *semantics* of the KB or, in other words, all the possible worlds that are compatible with the information it represents.

The main task of a system based on the KB paradigm is to answer user-specified *queries* (Calvanese et al. 2007b; Bienvenu and Ortiz 2015; Calì, Gottlob, and Kifer 2013). In simple words, a *query* is a formal expression describing the information the user wants to retrieve. *Answering* a query over a KB amounts to extracting the information specified by the former from the models of the latter and represent such information in a way that can be presented to users. These representations are usually called *answers*.

Example 1. *Assume logical predicates* $\mathsf{Employee}_{/1}(\mathsf{E})$, $\mathsf{hasSupervisor}_{/2}(\mathsf{hS})$. *The KB* $\mathcal{K}$ *consists of the TBox* $\mathcal{T}$ *and the ABox* $\mathcal{D}$ *defined as follows:*

$$\begin{array}{lllll} \mathcal{T} = & \{\mathsf{E} \sqsubseteq \exists \mathsf{hS} & \exists \mathsf{hS}^{-} \sqsubseteq \mathsf{E} & \mathsf{hS} \sqsubseteq \neg \mathsf{hS}^{-}\} \\ \mathcal{D} = & \{\mathsf{E}(Ava), & \mathsf{E}(Bea), & \mathsf{hS}(Bea, Carl)\} \end{array}$$

Intuitively, $\mathcal{K}$ *describes an organization where employees are assigned to supervisors that are themselves employees. Moreover, employees cannot supervise their supervisors, i.e.,* hS *is anti-symmetric and irreflexive. Formally, the models of* $\mathcal{K}$ *are all the first-order interpretations* $\mathcal{I}$ *that satisfy* $\mathcal{T}$ *for which* $Ava^{\mathcal{I}}, Bea^{\mathcal{I}} \in \mathsf{E}^{\mathcal{I}}$ *and* $(Bea^{\mathcal{I}}, Carl^{\mathcal{I}}) \in \mathsf{hS}^{\mathcal{I}}$.

Suppose we want to retrieve from $\mathcal{K}$ *information about employees and their supervisors. This can be done with the following first-order logic query (in fact, conjunctive query):*

$$q = \{(x, y) \mid \mathsf{hS}(x, y)\}$$

An answer for q *over* $\mathcal{K}$ *could be* $(Bea, Carl)$*, since* $Carl^{\mathcal{I}}$ *is the supervisor of* $Bea^{\mathcal{I}}$ *in every model* $\mathcal{I}$ *of* $\mathcal{K}$.

Carefully choosing the right notion of answers for a specific application domain is crucial to obtain meaningful information from a KB. This is because query answers must satisfy two seemingly antithetic needs. On the one hand, they should represent as much as possible the information that users require with their queries. On the other, they must be simple and compact enough to be reasonably understood and computed. Whenever a KB has only one single and finite model, these requirements can be met simply by returning all those data tuples that satisfy the query expression over such model (Abiteboul, Hull, and Vianu 1995). Under reasonable assumptions, this set is finite and represents all the information defined by the query and the KB. In all the other cases, we need more sophisticated answers.

To answer queries posed to KBs with multiple or even infinite models, the majority of approaches proposed in the literature fall in one of two families: *certain answers* or *possible answers* (see (Lipski 1979) for an early account). Intuitively, the former are constants that answer a given query in *every model of the KB*, while the latter answer the query *in at least one of these models*. Despite their wide adoption, both certain and possible answers cannot represent all the information that a query requires in several applications (Console, Guagliardo, and Libkin 2016, 2022; Calautti, Console, and Pieris 2021).

Example 2. *Consider again the scenario of Example 1. It is easy to verify that the set* Θ *of certain answers for* q *over* $\mathcal{K}$ *is* $\{(Bea, Carl)\}$ *while the set* Π *of possible answers for*

q over $\mathcal{K}$ contains every pair of distinct constants except $(Carl, Bea)$. Both Θ and Π, however, do not represent all the information that q can extract from all the models of $\mathcal{K}$.

For example, Θ loses the information that, for every model $\mathcal{I}$ of $\mathcal{K}$, there exist constants m and s such that both *$(Ava^{\mathcal{I}}, s)$ and (s, m) satisfy q in $\mathcal{I}$. Similarly, Π loses the information that, for every model $\mathcal{I}$ of $\mathcal{K}$ and every pair of constants (e, s), either (e, s) or (s, e) do not satisfy q in $\mathcal{I}$.*

Issues related to the informativeness of query answers are well-known in incomplete databases (Console et al. 2020). The main challenge here is to define *representation systems*, i.e., families of syntactic objects that can represent the desired set of answers under some requirement of informativeness (Abiteboul, Hull, and Vianu 1995). Crucially, for representation systems to work, syntactic objects require semantics, i.e., a formal specification of the set of answers they represent.

Representation systems exist in different flavors. A prominent notion in this context is that of *strong representation systems* (SRS), i.e., objects whose semantics coincide exactly with the set of answers we want to represent. SRS, however, are notoriously hard to obtain, and require complex representations that are hard to understand for a user. For this reason, (Imielinski and Lipski 1984) proposed the notion of *weak representation systems* (WRS), where informativeness requirements are formulated in terms of a semantic equivalence relation based on certain knowledge. While their scope of application is broader, WRS are still tight to a specific notion of equivalence. For this reason, (Libkin 2016) introduced the framework of *representation systems* that generalizes SRS and WRS by allowing a more general notion of equivalence. The same framework has been used in (Civili and Libkin 2018) to define approximations of certain answers for expressive KB queries.

In the field of KBs, where queries often fall into simple fragments of first-order logic to avoid undecidability issues, the scarce informativeness of certain and possible answers is partly mitigated by the simple form that queries can take. Despite these simplifications, issues of informativeness are well-known in the literature. For example, in the context of *data exchange*, (Arenas, Pérez, and Reutter 2013) defines universal solutions via SRS while (Grahne and Onet 2012) uses them for query answering purposes . In the Description Logics setting, the work in (Borgida, Toman, and Weddell 2016, 2017) argues that certain answers cannot represent crucial information requested by queries and suggest the use of *referring expression* to mitigate the issue. In simple words, referring expressions are first-order sentences that further describe the properties of constants in the answers. In a similar setting, the work in (Lutz and Przybylko 2022, 2023) suggests the use of *minimal partial answers*, i.e., tuples with constant and variable symbols called *wildcards*. Crucially, however, wildcards are allowed to repeat only within tuples but not across them. This idea, akin to the V-Tables explored in (Imielinski and Lipski 1984), can represent more information than certain answers but it is still unsatisfactory in general.

Example 3. *Consider, once again, the setting of Example 1. The set C_W of minimal partial answers for q over $\mathcal{K}$ consist of the tuples $(Ava, \star)$, $(Carl, \star')$, and $(Bea, Carl)$, where $\star$ and $\star'$ are wildcards. Clearly, C_W is much more informative than the set $\Theta = \{(Bea, Carl)\}$ of certain answers for q over $\mathcal{K}$. However, C_W still does not capture the information described in Example 2. Conversely, the following set C_V, still based on V-Tables, captures such information: $C_V = \{(Ava, \star_1), (\star_1, \star_2), (Bea, Carl), (Carl, \star'_1), (\star'_1, \star'_2)\}$, where $\star_1$, $\star_2$, $\star'_1$, and $\star'_2$ are variables.*

To the best of our knowledge, there is no formal framework in the literature that can characterize the informative content of query answers for KBs. Thus, our intuition that, in Example 3, C_V is more informative than C_W cannot be formally proved. The main goal of this paper is to fill this gap, and use this novel framework to define more informative answers for queries over KBs.

More specifically, the contribution of the paper is the following. Firstly, we present a framework to characterize the information that query answers for KBs are able to represent. Our framework is inspired by the notion of representation systems presented in (Libkin 2016), with suitable modifications to work with KBs. As a first application of our framework, we study the informativeness of current query answering approaches and define properties of answer objects that are needed for more informative results. Such results would be difficult to obtain without a framework of this kind and, thus, we consider them a central contribution of this work. Then, we introduce a novel notion of answers based on V-tables. This notion, similar in spirit to the one used in (Lutz and Przybylko 2022, 2023), allows repetition of variables across answer tuples. Using our framework, we prove that these answers are able to represent a meaningful form of information whenever KBs possess widely accepted structural properties and queries are expressed as unions of conjunctive queries. Finally, we study the computational characteristics of these answers for KBs expressed in well-known languages for formulating KBs.

The remainder of this paper is organized as follows. In Section 2 we introduce notions from the literature that will be used in this work. In Section 3 we present our framework for informativeness of query answers, and in Section 4 we study current techniques for query answering under the lens of our framework. In Section 5 we present novel forms of answers for KBs and study their informativeness, while their computational properties are studied in Section 6. Finally, in Section 7 we conclude.

2 Preliminaries

In what follows, we fix two countably infinite and disjoint sets $\mathcal{C}$ and $\mathcal{V}$ used for *constants* and *variables*, respectively.

First-Order Logic. Due to space limitations, we assume that the reader is familiar with the syntax and semantics of first-order logic and refer to (Levesque and Lakemeyer 2001) for a detailed account. A *signature* is a countably infinite set of predicate symbols disjoint from $\mathcal{C} \cup \mathcal{V}$. Let $\mathcal{S}$ be a signature, an *atom over $\mathcal{S}$* is a syntactic expression of the form $p(t_1, \ldots, t_m)$, where $p \in \mathcal{S}$ is a predicate symbol of arity m and $t_i \in \mathcal{C} \cup \mathcal{V}$, for each $i = 1, \ldots, m$. An atom

is *ground* if it contains not variables. Moreover, we will use $\mathbf{FO}(\mathcal{S})$ to denote the set of all the *first-order (FO) sentences* that can be constructed from symbols of $\mathcal{S}$, $\mathcal{C}$, and $\mathcal{V}$ for predicate names, constants, and variables, respectively.

In the technical development of this paper, formulae will be interpreted under the *standard name assumption* using constants in $\mathcal{C}$ as standard names. This is done for the sake of a simplicity but all the results we present can be easily extended to the general case. Formally, an *interpretation for* $\mathcal{S}$ is a pair $\mathcal{I} = \langle \mathcal{C}, \cdot^{\mathcal{I}} \rangle$ such that $\cdot^{\mathcal{I}}$ assigns to each constant $c \in \mathcal{C}$ itself, i.e. $c^{\mathcal{I}} = c$ for each $c \in \mathcal{C}$, and to each n-ary predicate $P \in \mathcal{S}$ an n-ary relation $P^{\mathcal{I}} \subseteq \mathcal{C}^n$. As usual, we will write $\mathcal{I} \models \varphi$ (resp., $\mathcal{I} \not\models \varphi$) to denote the fact that $\mathcal{I}$ satisfies (resp., does not satisfy) $\varphi \in \mathbf{FO}(\mathcal{S})$. Moreover, for readability purposes, we will often write $P(\bar{c}) \in \mathcal{I}$ (resp. $P(\bar{c}) \notin \mathcal{I}$) for $\bar{c} \in P^{\mathcal{I}}$ (resp. $\bar{c} \notin P^{\mathcal{I}}$) and use $\subseteq$ and $\not\subseteq$ accordingly. In this way, we will effectively blur the distinction between sets of ground atoms and interpretations.

Relevant Fragments of FO. We proceed to define fragments of **FO** that are relevant for the technical development of this paper. Given $\mathcal{S}$ as above, a *language of sentences* (simply, language) over $\mathcal{S}$ is a subset of $\mathbf{FO}(\mathcal{S})$. The language $\exists\mathsf{Pos}(\mathcal{S})$ consists of all the sentences in $\mathbf{FO}(\mathcal{S})$ of the form $\bigvee_i \exists \bar{x}_i.\, \varphi_i(\bar{x}_i)$ where each $\varphi_i(\bar{x}_i)$ is a conjunction of atoms over $\mathcal{S}$ with variables in $\bar{x}_i$. We will use $\exists\mathsf{Pos}_{local}(\mathcal{S})$ for the fragment of $\exists\mathsf{Pos}(\mathcal{S})$ where variables are not allowed to repeat across atoms. Other languages of interest are defined by restricting the number of variables allowed in their sentences. For every positive integer n, we use $\exists\mathsf{Pos}[n](\mathcal{S})$ for the language of sentences in $\exists\mathsf{Pos}(\mathcal{S})$ of the form $\bigvee_i \exists \bar{x}_i.\, \varphi_i(\bar{x}_i)$ where each $\bar{x}_i$ consists of *at most* n *distinct variables* and $\exists\mathsf{Pos}_v[n](\mathcal{S})$ for the restriction of $\exists\mathsf{Pos}[n](\mathcal{S})$ to *constant-free sentences*. Finally, we will use $\exists\mathsf{Pos}_G(\mathcal{S})$ for the restriction of $\exists\mathsf{Pos}(\mathcal{S})$ to sentences *without variables* and $\mathsf{GA}(\mathcal{S})$ for *ground atoms*.

Valuations and Homomorphisms. Valuations and homomorphisms will be crucial in the technical development of this work. A *valuation* v is simply a function from $\mathcal{C} \cup \mathcal{V}$ to $\mathcal{C} \cup \mathcal{V}$ such that $v(c) = c$ for each $c \in \mathcal{C}$. Given an atom $\alpha = R(t_1, \ldots, t_m)$ and a valuation v, we write $v(\alpha)$ for the atom $R(v(t_1), \ldots, v(t_m))$ and extend this notation to sets of atoms in the obvious way. Given two sets of atoms over A and B, we say that a valuation h is a *homomorphism* if, for every $\alpha \in A$, we have $h(\alpha) \in B$.

Knowledge Bases. We fix a signature Σ and call its elements *ontological predicates*. In this paper, whenever we refer to a *knowledge base (KB)* $\mathcal{K}$, we implicitly mean a pair $\langle \mathcal{T}, \mathcal{D} \rangle$, where $\mathcal{T}$ is a finite set of sentences in $\mathbf{FO}(\Sigma)$ that do not mention constants and $\mathcal{D}$ is a finite set of sentences in $\mathsf{GA}(\Sigma)$. Usually, $\mathcal{T}$ is called the TBox of $\mathcal{K}$ while $\mathcal{D}$ is called the ABox of $\mathcal{K}$ (Baader et al. 2003). We denote by $\mathcal{C}(\mathcal{D})$ the constants appearing in $\mathcal{D}$. As usual, the semantics of a KB $\mathcal{K}$ is given in terms of its *models*, which are those interpretations for Σ that satisfy the assertions of $\mathcal{K}$. Given a KB $\mathcal{K}$, we denote by $\mathsf{Mod}(\mathcal{K})$ the set of its models, and say that $\mathcal{K}$ is *consistent* if $\mathsf{Mod}(\mathcal{K}) \neq \emptyset$; *inconsistent* otherwise.

A *KB language* $\mathbb{K}$ is simply a family of KBs, and a KB $\mathcal{K}$ will be a called an $\mathbb{K}$ KB if it belongs to $\mathbb{K}$. In what follows, we will be interested in several KB languages defined either by structural properties or by syntactic definitions. We say that a KB language $\mathbb{K}$ enjoys the *universal model property* if, for every consistent $\mathbb{K}$ KB $\mathcal{K}$, there exists a set $\mathcal{U}$ of atoms over Σ and terms from $\mathcal{C}(\mathcal{D}) \cup \mathcal{V}$ for predicate names and terms, respectively, satisfying the following properties:

- for every $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$ there exists a homomorphism from $\mathcal{U}$ to $\mathcal{I}$ (seen as a set of atoms); and
- for every bijection $v : \mathcal{V} \to (\mathcal{C} \setminus \mathcal{C}(\mathcal{D}))$, $v(\mathcal{U})$ (seen as an interpretation) belongs to $\mathsf{Mod}(\mathcal{K})$.

We use *Univ* for the language of consistent KBs with the universal model property. Furthermore, we say that a KB language $\mathbb{K}$ enjoys the *invariance under disjoint-union property* if, for every consistent KB $\langle \mathcal{T}, \mathcal{D} \rangle$ in $\mathbb{K}$, and every finite set of ground atoms $\mathcal{B}$ such that $\langle \mathcal{T}, \mathcal{B} \rangle$ is consistent and $\mathcal{B}$ does not use constants in $\mathcal{D}$, also $\langle \mathcal{T}, \mathcal{D} \cup \mathcal{B} \rangle$ is consistent. We use *Disj* for the language of consistent KBs with the invariance under disjoint-union property. Many expressive KB languages enjoy such properties, e.g. tuple-generating dependencies (tgds) and the Description Logic (DL) $\mathcal{ALCHI}$.

Finally, we mention that our negative results will be often formulated with respect to the KB language *DL-Lite*$_{core}$, which is the least expressive KB language of the lightweight *DL-Lite* family (Calvanese et al. 2007b) of DLs.

Queries Whenever we will refer to a *query* q, we implicitly mean an expression of the form $q = \{\bar{x} \mid \varphi(\bar{x})\}$, where $\varphi(\bar{x})$ is a first-order formula whose free variables are those occurring in $\bar{x}$ and which adopts symbols from Σ for predicate names, symbols from $\mathcal{C}$ for constants, and symbols from $\mathcal{V}$ for variables. The *arity* of a query q as above is the arity of the tuple $\bar{x}$, and a query q is called n-ary if n is its arity. We denote by FQ the language of all the possible (FO) queries.

Given an interpretation $\mathcal{I}$ for Σ and an n-ary query $q = \{\bar{x} \mid \varphi(\bar{x})\}$, an n-tuple $\bar{c} \in \mathcal{C}^n$ of constants is an *answer tuple for* q *over* $\mathcal{I}$ if $\mathcal{I}$ satisfies the $\mathbf{FO}(\Sigma)$ sentence $\varphi(\bar{x}/\bar{c})$, i.e. if it holds that $\mathcal{I} \models \varphi(\bar{x}/\bar{c})$. For a Boolean query $q = \{() \mid \varphi\}$ and an interpretation $\mathcal{I}$ for $\Sigma \cup \mathcal{C}$, we say that $\mathcal{I}$ satisfies q if $\mathcal{I} \models \varphi$. Given a KB $\mathcal{K}$ and an n-ary query q, an n-tuple $\bar{c} \in \mathcal{C}^n$ of constants is a *certain (resp. possible) answer tuple for* q *over* $\mathcal{K}$ if $\mathcal{I} \models \varphi(\bar{x}/\bar{c})$, for every (resp. some) model $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$. For simplicity, given a Boolean (i.e. 0-ary) query $q = \{() \mid \varphi\}$ and a KB $\mathcal{K}$, we say that $\mathcal{K}$ entails q if $\mathcal{I} \models \varphi$ for every model $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$.

In the technical sections of this paper, we are interested in *conjunctive queries* and unions thereof, whose languages will be denoted by CQ and UCQ, respectively. Formally, a union of conjunctive queries (UCQ) is a query of the form $q = \{\bar{x} \mid \exists \bar{y}_1.\phi_1(\bar{x}, \bar{y}_1) \vee \ldots \vee \exists \bar{y}_m.\phi_m(\bar{x}, \bar{y}_m)\}$ such that $\phi_i(\bar{x}, \bar{y}_i)$ is a conjunction of atoms for each $i = 1, \ldots, m$, and q is also a conjunctive query (CQ) if $m = 1$.

3 Framework

We now introduce our framework to characterize the information content of query answers for KBs. To this end, we first provide a general definition of query answers that can accommodate different scenarios. Assume a query language $\mathbb{Q}$ and a KB language $\mathbb{K}$. A *query evaluation function* for $\mathbb{Q}$ over $\mathbb{K}$ is simply a function $eval$ from $\mathbb{Q} \times \mathbb{K}$ to a set of objects $\mathcal{A}$. Intuitively, $eval$ describes the behavior of a query

evaluation system, and the elements of $\mathcal{A}$, called the *answer objects* of $eval$, are understood as the answers that such system provides. The goal of our framework is to characterize how informative these objects are.

To this end we observe that, in principle, query evaluation functions allow for any kind of answer objects. However, in order to be meaningful, query answers should represent information that queries actually require. To formally establish this connection, our framework provides a mechanism to connect the answer objects returned by a query evaluation function for $\mathbb{Q}$ over $\mathbb{K}$ to the information that queries of $\mathbb{Q}$ extract from the models of KBs of $\mathbb{K}$. Our next step is to formalize such mechanism.

Let $\mathsf{ANS} = \bigcup_{i=0}^{\infty}\{\mathsf{ans}_i\}$ be a countably infinite set of predicates where each ans_i has arity i and does not occur in Σ. An *answer structure* is simply an interpretation for ANS, and we denote *the set of all answer structures* by $\mathsf{ANS}^{\mathcal{C}}$. Given an n-ary query q, the *complete answer for q over an interpretation $\mathcal{I}$* (written $q(\mathcal{I})$) is the answer structure such that $\mathsf{ans}_n(\bar{a}) \in q(\mathcal{I})$ iff $\bar{a}$ is an answer tuple for q over $\mathcal{I}$ and $\mathsf{ans}_m^{q(\mathcal{I})} = \emptyset$, for each $m \neq n$. Intuitively, $q(\mathcal{I})$ collects all the answer tuples for q over $\mathcal{I}$ within a single structure. This rather standard construction allows to define the information provided by the answer tuples for q over $\mathcal{I}$ *in terms of logically-definable properties*.

Example 4. *Assume predicates $R_{/2}, S_{/2}$, the interpretation $\mathcal{I}$ such that $R^{\mathcal{I}} = \{(a,b),(c,d)\}$ and $S^{\mathcal{I}} = \{(b,c),(d,b)\}$, and the query $q = \{(x,y) \mid \exists z.R(x,z) \wedge S(z,y)\}$. Then, $ans_2^{q(\mathcal{I})} = \{(a,c),(c,b)\}$, and it is easy to see the following:*

- *$q(\mathcal{I})$ contains a path of length 2, i.e., $q(\mathcal{I}) \models \psi_1$ where $\psi_1 = \exists x,y,z.\, ans_2(x,y) \wedge ans_2(y,z)$;*
- *$q(\mathcal{I})$ does not contain (an embedding of) a triangle, i.e., $q(\mathcal{I}) \not\models \psi_2$ where $\psi_2 = \exists x,y,z.\, ans_2(x,y) \wedge ans_2(y,z) \wedge ans_2(z,x)$;*
- *$q(\mathcal{I})$ is not symmetric, i.e., $q(\mathcal{I}) \not\models \psi_3$ where $\psi_3 = \forall x,y.\, ans_2(x,y) \rightarrow ans_2(y,x)$;*

Complete answers can be extended to KBs in the natural way. Given a KB $\mathcal{K}$, the *set of complete answers for q over $\mathcal{K}$*, written $q(\mathcal{K})$, is defined as follows:

$$q(\mathcal{K}) = \{q(\mathcal{I}) \mid \mathcal{I} \in \mathsf{Mod}(\mathcal{K})\}$$

Simply put, $q(\mathcal{K})$ collects all the complete answers that one can obtain from q and the models of $\mathcal{K}$.

Example 5. *Assume the scenario of Example 4 and let $\mathcal{J}$ be such that $R^{\mathcal{J}} = \{(e,f),(g,h)\}$ and $S^{\mathcal{J}} = \{(f,g),(h,f)\}$. Finally, let $\mathcal{K}$ be a KB such that $\mathsf{Mod}(\mathcal{K}) = \{\mathcal{I},\mathcal{J}\}$. Then, $q(\mathcal{K}) = \{q(\mathcal{I}), q(\mathcal{J})\}$ with $ans_2^{q(\mathcal{J})} = \{(e,g),(g,h)\}$.*

Clearly, $q(\mathcal{K})$ contains all the information that q extracts from the models of $\mathcal{K}$. To establish a meaningful semantics for an answer object $eval(q,\mathcal{K})$, then, we provide a mechanism to define the properties of the elements of $q(\mathcal{K})$, i.e., answer structures, that are represented by $eval(q,\mathcal{K})$. Formally, an *answer domain* is a pair $\langle \mathcal{A}, \models_A \rangle$, where $\mathcal{A}$ is a set of objects and $\models_A \subseteq \mathcal{A} \times \mathbf{FO}(\mathsf{ANS})$ is a relation. In simple words, the goal of answer domains is to provide formal semantics to a set $\mathcal{A}$ of answer objects. This is done by the relation $\models_A$ that defines the (first-order definable) properties of answer structures the objects of $\mathcal{A}$ satisfy. The notion of *query answering system*, defined next, connects query evaluation functions and answer domains.

Definition 1. *A query answering system (QAS) is a tuple $\langle \mathbb{Q}, \mathbb{K}, \mathbb{A}, \mathsf{eval} \rangle$, where $\mathbb{Q}$ is a query language, $\mathbb{K}$ is a KB language, $\mathbb{A} = \langle \mathcal{A}, \models_A \rangle$ is an answer domain, and eval is a query evaluation function from $\mathbb{Q} \times \mathbb{K}$ to $\mathcal{A}$.*

In the next example, we define a QAS that behaves according to the classical notion of certain answers.

Example 6. *Let $\mathbb{K}$ be an arbitrary KB language and let $\mathbb{S}_G = \langle \mathsf{FO}, \mathbb{K}, \mathbb{A}_G, \mathsf{cert}_G \rangle$ be the QAS defined as follows:*

- *$\mathbb{A}_G = \langle 2^{\mathsf{GA}(\mathsf{ANS})}, \models_G \rangle$, where $A \models_G \varphi$ if and only if A (seen as an interpretation) satisfies φ, i.e., $A \models \varphi$; and*
- *$\mathsf{cert}_G(q,\mathcal{K})$ is the set of all $ans_p(\bar{c})$, where p is the arity of q and $\bar{c}$ is a certain answer tuple for q over $\mathcal{K}$.*

Let q, ψ_1, ψ_2, and ψ_3 be as in Example 4. We have that $\mathsf{cert}_G(q,\mathcal{K}) \models_G \psi_1$ iff the set of certain answers for q over $\mathcal{K}$ contains a path of length 2, $\mathsf{cert}_G(q,\mathcal{K}) \models_G \psi_2$ iff such set contains (an embedding of) a triangle, and $\mathsf{cert}_G(q,\mathcal{K}) \models_G \psi_3$ iff such set is symmetric.

With the notion of QAS in place, we are now ready to characterize the information content of answer objects. Specifically, for a QAS $\langle \mathbb{Q}, \mathbb{K}, \mathbb{A}, \mathsf{eval} \rangle$, our characterization is based on the (FO definable) properties of $q(\mathcal{K})$ that $eval(q,\mathcal{K})$ represents according to $\mathbb{A}$. In what follows, we are interested in two families of properties: those that hold in $q(\mathcal{I})$ for every (resp., some) $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$. Intuitively, these properties represent the certain (resp., possible) knowledge provided by $q(\mathcal{K})$. Clearly, other interesting family of properties exist but they go beyond the scope of our work.

Formally, we define the relations $\models_c$ and $\models_p$ as follows:

$$\models_c \equiv \{(B,\varphi) \mid B \subseteq \mathsf{ANS}^{\mathcal{C}} \text{ and } b \models \varphi, \text{for each } b \in B\}$$

$$\models_p \equiv \{(B,\varphi) \mid B \subseteq \mathsf{ANS}^{\mathcal{C}} \text{ and } b \models \varphi, \text{for some } b \in B\}$$

Definition 2. *Let $\mathbb{S} = \langle \mathbb{Q}, \mathbb{K}, \langle \mathcal{A}, \models_A \rangle, \mathsf{eval} \rangle$ be a QAS and $\mathcal{F} \subseteq \mathbf{FO}(\mathsf{ANS})$ be a language of formulae. We say that*

- *$\mathbb{S} \models_c$-preserves $\mathcal{F}$ if, for each $q \in \mathbb{Q}$, $\mathcal{K} \in \mathbb{K}$, and $\varphi \in \mathcal{F}$, it holds that $\mathsf{eval}(q,\mathcal{K}) \models_A \varphi$ if and only if $q(\mathcal{K}) \models_c \varphi$;*
- *$\mathbb{S} \models_p$-preserves $\mathcal{F}$ if, for each $q \in \mathbb{Q}$, $\mathcal{K} \in \mathbb{K}$, and $\varphi \in \mathcal{F}$, it holds that $\mathsf{eval}(q,\mathcal{K}) \models_A \varphi$ if and only if $q(\mathcal{K}) \models_p \varphi$.*

The technical tool provided by Definition 2 allows to characterize and compare the information content of query answers. Specifically, the larger the language $\mathcal{F}$ that a QAS $\mathbb{S} \models_c$-preserves (resp, $\models_p$-preserves), the more informative such a QAS is w.r.t. those properties. The tool that this definition provides represents the heart of our framework.

Example 7. *Let $\mathbb{S}_G$ as in Example 6 and assume that $\mathbb{K}$ is the family of DL-Lite$_{core}$ knowledge bases. If $\mathsf{cert}_G(q,\mathcal{K}) \models_G \alpha$, for some $\alpha \in \mathsf{GA}(\mathsf{ANS})$, then, $A \models \alpha$, for each $A \in q(\mathcal{K})$. Thus, we can conclude that $\mathbb{S}_G$ preserves the certain knowledge of $\mathsf{GA}(\mathsf{ANS})$. However, $\mathbb{S}_G$ does not preserve the possible knowledge of $\mathsf{GA}(\mathsf{ANS})$. This is because, intuitively, if $\mathsf{cert}_G(q,\mathcal{K}) \not\models_G \alpha$, for some $\alpha \in \mathsf{GA}(\mathsf{ANS})$, there may still exist $A \in q(\mathcal{K})$ such that $A \models \alpha$.*

QAS can define several different notions of query answers for the same KB and query languages. However, one can show that QAS that preserve (either $\models_p$ or $\models_c$) a given language $\mathcal{F}$ are equivalent w.r.t. $\mathcal{F}$ in the following sense. Let $\mathbb{S}_1 = \langle \mathbb{Q}, \mathbb{K}, \langle \mathcal{A}_1, \models_1 \rangle, \mathsf{eval}_1 \rangle$ and $\mathbb{S}_2 = \langle \mathbb{Q}, \mathbb{K}, \langle \mathcal{A}_2, \models_2 \rangle, \mathsf{eval}_2 \rangle$ be two QAS and $\mathcal{F} \subseteq \mathbf{FO}(\mathsf{ANS})$ be a language of sentences. We say that $\mathbb{S}_1$ *and* $\mathbb{S}_2$ *are* $\mathcal{F}$*-equivalent* if $\mathsf{eval}_1(q, \mathcal{K}) \models_1 \varphi$ if and only if $\mathsf{eval}_2(q, \mathcal{K}) \models_2 \varphi$, for every $q \in \mathbb{Q}$, $\mathcal{K} \in \mathbb{K}$, and $\varphi \in \mathcal{F}$.

Proposition 1. *Let $\mathbb{S}_1$ and $\mathbb{S}_2$ be QAS that $\models_\star$-preserve $\mathcal{F}$ (with $\star \in \{c, p\}$). We have that $\mathbb{S}_1$ and $\mathbb{S}_2$ are $\mathcal{F}$-equivalent.*

Answers as Sets of Tuples Obviously, QAS can accommodate several forms of query answers for KBs, from very simple to very complex. However, while very expressive answers paired with suitable evaluation functions may be able to preserve more information, understanding these representations may become a very challenging task. In light of Proposition 1, we will focus on QAS based on a simple form of answers akin to the classical V-Tables. As we will show, these answers are expressive enough to preserve a meaningful family of properties.

Let $\mathcal{A}_V$ be the family of all sets (finite or infinite) of atoms over ANS, $\mathcal{C}$, and $\mathcal{V}$. Given $A \in \mathcal{A}_V$, *the semantics of* A, written $[\![A]\!]$, is the set of all the answer structures obtained from A by replacing variables with constants. We define $\models_V \subseteq \mathcal{A}_V \times \mathbf{FO}(\mathsf{ANS})$ such that, for each $A \in \mathcal{A}_V$, $A \models_V \varphi$ if and only if $C \models \varphi$, for each $C \in [\![A]\!]$.

Definition 3. *$\mathbb{A}_{tup}$ is the answer domain $\langle \mathcal{A}_V, \models_V \rangle$.*

We say that a QAS $\mathbb{S} = \langle \mathbb{Q}, \mathbb{K}, \mathbb{A}, \mathsf{eval} \rangle$ is *based on tuple-set answers* if $\mathbb{A} = \mathbb{A}_{tup}$ as in Definition 3. Moreover, we say that $\mathbb{S}$ is based on *finite tuple-set answers* if it is based on tuple-set answers and, additionally, the set $\mathsf{eval}(q, \mathcal{K})$ is finite, for every $q \in \mathbb{Q}$ and $\mathcal{K} \in \mathbb{K}$.

In the remainder of the paper, we will study QAS based on tuple-set answers and show that, in some cases, this seemingly simple family is able to preserve relevant part of possible and certain knowledge. Before concluding this section, we observe that the definition of $\models_V$ is intuitively based on the *closed-world assumption* for the elements of $\mathbb{A}_V$. Obviously, other definitions are possible, e.g., using the *open-world* or *weak closed-world* assumptions. However, due to the languages considered, any of these three definitions would not make any difference and, thus, we opted for the one that, we believe, yields the clearest construction.

4 Current Approaches to Query Answering

As a first application of our framework, we study the information content of query answers from the literature and discuss properties that are needed for more informative QAS. Besides being an essential first step towards the definition of more informative query answers, results in this section shed light on techniques that are widely adopted in the literature and, thus, they are interesting in their own right.

Firstly, we proceed to analyze the classical notion of certain answers. To this end, we define the QAS $\mathbb{C}_u = \langle \mathsf{UCQ}, \mathit{Univ}, \mathbb{A}_{tup}, \mathsf{cert}_G \rangle$ where cert_G is the evaluation function defined in Example 6. Intuitively, $\mathbb{C}_u$ captures the case of classical certain answers for UCQ over *Univ* KBs.

Proposition 2. *$\mathbb{C}_u \models_c$-preserves $\exists\mathsf{Pos}_G(\mathsf{ANS})$.*

Proposition 2 tells us that, as expected, cert_G captures all the certain information about properties that can be expressed as conjunctions of ground atoms. Unsurprisingly, though, this property ceases to hold if we consider languages slightly beyond $\exists\mathsf{Pos}_G(\mathsf{ANS})$. Let $\mathbb{C}_{\text{DL}} = \langle \mathsf{CQ}, \mathit{DL\text{-}Lite}_{core}, \mathbb{A}_{tup}, \mathsf{cert}_G \rangle$. The case of $\mathbb{C}_{\text{DL}}$ is already problematic, as the following proposition shows.

Proposition 3. *$\mathbb{C}_{DL}$ does not $\models_c$-preserve $\exists\mathsf{Pos}[1](\mathsf{ANS})$, and it does not $\models_c$-preserve $\exists\mathsf{Pos}_{local}(\mathsf{ANS})$.*

We now turn our attention to an extended form of certain answers that was recently presented in (Lutz and Przybylko 2022, 2023). To define a suitable QAS for these answers, we need to introduce some additional notation. Let $\mathcal{K}$ be a KB and q a query. We say that an atom α over ANS is a *partial answer for q over $\mathcal{K}$* if, for every $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$, there is a homomorphism from $\{\alpha\}$ to $q(\mathcal{I})$ (seen as a set of atoms). Let now β be an atom over ANS. We say that α is *more informative than* β (written $\beta \prec \alpha$) if there is a homomorphism h from $\{\beta\}$ to $\{\alpha\}$ but not vice versa. Finally, we say that α *is a minimal partial answer for q over $\mathcal{K}$* if it is a partial answer for q over $\mathcal{K}$ and there exists no β such that β is partial answer for q over $\mathcal{K}$ and $\alpha \prec \beta$.

Definition 4. *Let $W = \{\alpha_1, \ldots, \alpha_m\}$ be a set of minimal partial answers for q over $\mathcal{K}$ that share no variables. The set W is a certain answer with wildcards for q over $\mathcal{K}$ if, for every minimal partial answer β for q over $\mathcal{K}$, there is $j \in [m]$ and a valuation v such that $v(\beta) = \alpha_j$.*

Intuitively, a certain answer with wildcards collects all the minimal partial answers for q over $\mathcal{K}$, up to variable renaming. We define the QAS associated to certain answers with wildcards as follows. Let cert_W be the query evaluation function that associates, to every pair of KB $\mathcal{K}$ and query q, a certain answer with wildcards for q over $\mathcal{K}$. We define the QAS $\mathbb{W}_{\text{DL}} = \langle \mathsf{CQ}, \mathit{DL\text{-}Lite}_{core}, \mathbb{A}_{tup}, \mathsf{cert}_W \rangle$.

Proposition 4. *$\mathbb{W}_{DL}$ does not $\models_c$-preserve $\exists\mathsf{Pos}[1](\mathsf{ANS})$.*

With respect to classical certain answers, the additional information that certain answers with wildcards are able to represent can be characterized by a rather specific language of sentences, as the following proposition shows.

Proposition 5. *$\mathbb{W}_{DL} \models_c$-preserves $\exists\mathsf{Pos}_{local}(\mathsf{ANS})$.*

Finally, we turn our attention to the other major family of answers for KBs, i.e., possible answers. To this end, we define $\mathbb{P}_{\text{DL}} = \langle \mathsf{CQ}, \mathit{DL\text{-}Lite}_{core}, \mathbb{A}_{tup}, \mathsf{poss}_G \rangle$, where poss_G associates, to each pair $\langle q, \mathcal{K} \rangle$, the set of all the possible answer tuples for q over $\mathcal{K}$.

Proposition 6. *$\mathbb{P}_{DL} \models_p$-preserves $\mathsf{GA}(\mathsf{ANS})$, and it does not $\models_p$-preserves $\exists\mathsf{Pos}_G(\mathsf{ANS})$.*

The positive result in Proposition 6 can be easily extended to more expressive KB languages. However, it is easy to see that $\mathbb{P}_{\text{DL}}$ does not have finite answers and, thus, it cannot be implemented in practice. To mitigate this issue, possible answers are often restricted to constants that explicitly appear in the KB. Let poss_G^{fin} be the evaluation function that associates, to each pair $\langle q, \mathcal{K} \rangle$, the set of all the possible answers

for q over $\mathcal{K}$ whose constants explicitly occur in $\mathcal{K}$, and let $\mathbb{P}_{fin}$ be the QAS $\langle \mathsf{CQ}, \textit{DL-Lite}_{core}, \mathbb{A}_{tup}, \mathsf{poss}_G^{fin} \rangle$. This restriction has a severe impact on the informativeness of the answers, as the following proposition shows.

Proposition 7. *$\mathbb{P}_{fin}$ does not $\models_p$-preserve* $\mathsf{GA}(\mathsf{ANS})$.

Before concluding this section, we discuss properties of QAS that are required to preserve specific languages.

Let $\mathbb{S} = \langle \mathbb{Q}, \mathbb{K}, \langle \mathcal{A}, \models_A \rangle, \mathsf{eval} \rangle$ be a QAS and $\mathbb{A}_{tup} = \langle \mathcal{A}_V, \models_V \rangle$ as in Definition 3. We say that $\mathbb{S}$ has the *tuple-set property* for a language $\mathcal{F} \subseteq \mathbf{FO}(\mathsf{ANS})$ if, for every $\mathcal{K} \in \mathbb{K}$ and $q \in \mathbb{Q}$, there exists a $A_V \in \mathcal{A}_V$ such that $\mathsf{eval}(q, \mathcal{K}) \models_A \varphi$ if and only if $A_{tup} \models_V \varphi$, for every $\varphi \in \mathcal{F}$. The QAS $\mathbb{S}$ has the *finite tuple-set property* if, additionally, A_V is finite.

Intuitively, $\mathbb{S}$ has the (finite) tuple-set property if the answers it provides are equivalent to a (finite) set of tuples. Such hypothesis is at the core of several practical query answering systems for KBs and classical databases. We now study the impact that these two properties have on the informativeness of query answers. Let $\mathbb{S}_{\text{DL}} = \langle \mathsf{CQ}, \textit{DL-Lite}_{core}, \mathbb{A}, \mathsf{eval} \rangle$. Even for such simple languages, the finite tuple-set property is too restrictive.

Proposition 8. *If $\mathbb{S}_{DL}$ has the finite tuple-set property for* $\exists\mathsf{Pos}(\mathsf{ANS})$*, then $\mathbb{S}_{DL}$ does not $\models_c$-preserve* $\exists\mathsf{Pos}(\mathsf{ANS})$.

Intuitively, Proposition 8 tells us that answers based on finite sets of tuples cannot represent all the certain properties definable in $\exists\mathsf{Pos}$ that UCQs extract from KBs defined in the very simple *DL-Lite*$_{core}$ language.

The case of possible knowledge is even more problematic, as the following proposition shows.

Proposition 9. *If $\mathbb{S}_{DL}$ has the tuple-set property for* $\exists\mathsf{Pos}_G(\mathsf{ANS})$*, then $\mathbb{S}_{DL}$ does not $\models_p$-preserve* $\exists\mathsf{Pos}_G(\mathsf{ANS})$.

Proposition 9 rules out the possibility of obtaining answers based on sets of tuples (finite or infinite) that represent the possible knowledge definable in $\exists\mathsf{Pos}_G(\mathsf{ANS})$ (and, thus, of $\exists\mathsf{Pos}[n](\mathsf{ANS})$) that UCQs extract from KBs defined in the very simple *DL-Lite*$_{core}$ language.

Finally, note that the results of Proposition 8 and 9 apply to every QAS $\mathbb{S} = \langle \mathsf{CQ}, \mathbb{K}, \mathbb{A}, \mathsf{eval} \rangle$ with $\mathbb{K} \supseteq$ *DL-Lite*$_{core}$.

5 More Informative Answers for KBs

Several well-known notions of query answers for KBs are based on sets of tuples. Due to Propositions 8 and 9, however, we cannot expect such answers to represent all the properties definable in $\exists\mathsf{Pos}(\mathsf{ANS})$ that UCQs can extract from simple *DL-Lite*$_{core}$ ontologies. These results, however, leave open the possibility of obtaining QAS that preserve properties expressible in more limited languages. Here, we focus on properties in $\exists\mathsf{Pos}(\mathsf{ANS})$ that are expressible with a bounded number of variables and show that suitable QAS can be obtained for both certain and possible knowledge.

Before presenting our results, however, we need to introduce some additional definitions. Given a set of atoms A, the *Variable-Join Graph* $J(A) = \langle A, E \rangle$ of A is the graph whose vertexes are the elements of A and there is an edge between a and b whenever a and b share at least one variable. We say that a set of atoms is *Variable-Connected* if its Variable-Join Graph is connected.

Definition 5. *An n-Connected Answer is a Variable-Connected set of atoms over* ANS *with at most n variables.*

We first analyze the case of certain knowledge and show how n-connected answers can be used to define a family of QAS that preserve the certain knowledge of $\exists\mathsf{Pos}[n](\mathsf{ANS})$. Let $\mathcal{K}$ be a knowledge base and q a query. We say that a set of atoms A over ANS is *certain for q over $\mathcal{K}$* if, for every model $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$, there exists a constant-preserving homomorphism from A to $q(\mathcal{I})$ (seen as a set of atoms).

Definition 6. *Let $A = \bigcup_{i=1}^{m} A_i$, where $A_1, \ldots, A_m$ are n-connected answers that share no variables. The set A is a certain n-answer for q over $\mathcal{K}$ if:*

- *A_i is certain for q over $\mathcal{K}$, for each $i \in [m]$; and*
- *For every n-connected answer B that is certain for q over $\mathcal{K}$ there is $j \in [m]$ and a bijection h s.t. $h(B) = A_j$.*

Intuitively, a certain n-answer collects all the n-connected answers that are certain for q over $\mathcal{K}$, up to variable renaming. For instance, by referring to Example 3, C_V is a certain 2-answer for q over $\mathcal{K}$. We proceed to present crucial properties of this form of answers. Let n be a positive integer. The query evaluation function cert_n associates, to every pair of KB $\mathcal{K}$ and query q, a certain n-answer for q over $\mathcal{K}$. Let $\mathbb{C}_n = \langle \mathsf{UCQ}, \textit{Univ}, \mathbb{A}_{tup}, \mathsf{cert}_n \rangle$.

Theorem 1. *$\mathbb{C}_n \models_c$-preserves* $\exists\mathsf{Pos}[n](\mathsf{ANS})$.

We now turn our attention to possible knowledge and use n-connected answers to define a family of QAS that $\models_p$-preserves $\exists\mathsf{Pos}_v[n](\mathsf{ANS})$. Before delving into the technical details, we observe that the case of possible knowledge is even more delicate than the one of certain knowledge. In light of Proposition 9, in fact, we cannot expect to obtain a QAS that preserves the possible knowledge of $\exists\mathsf{Pos}[n](\mathsf{ANS})$, even for very simple KBs. However, tuple-set answers can still represent meaningful information in this setting with some additional restrictions.

Let $\mathcal{K}$ be a knowledge base and q a query. We say that a set of atoms A over ANS is *possible for q over $\mathcal{K}$* if there exists a model $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$ and a constant-preserving homomorphism from A to $q(\mathcal{I})$ (seen as a set of atoms).

Definition 7. *Let $A = \bigcup_{i=1}^{m} A_i$, where $A_1, \ldots, A_m$ are n-connected answers that share no variables and mention no constants. The set A is a possible n-answer for q over $\mathcal{K}$ if:*

- *A_i is possible for q over $\mathcal{K}$, for each $i \in [m]$; and*
- *For every n-connected answer B that is possible for q over $\mathcal{K}$ there is $j \in [m]$ and a bijection h s.t. $h(B) = A_j$.*

Intuitively, a possible n-answer collects all the constant-free n-connected answers that are possible for q over $\mathcal{K}$, up to variable renaming. For a positive integer n, let poss_n be a query evaluation function that associates, to every pair of knowledge base $\mathcal{K}$ and query q a possible n-answer for q over $\mathcal{K}$. We define the QAS $\mathbb{P}_n = \langle \mathsf{UCQ}, \textit{Disj}, \mathbb{A}_{tup}, \mathsf{poss}_n \rangle$.

Theorem 2. *$\mathbb{P}_n \models_p$-preserves* $\exists\mathsf{Pos}_v[n](\mathsf{ANS})$.

We observe that $\mathbb{C}_n$ and $\mathbb{P}_n$ are not the only QAS that preserve the certain and possible knowledge of $\exists\mathsf{Pos}[n](\mathsf{ANS})$. However, every such QAS are equivalent, in the sense of Proposition 1, respectively, to $\mathbb{C}_n$ and $\mathbb{P}_n$.

6 Computational Aspects

In this section, we investigate some relevant decision problems associated with the query answering problem for the newly introduced certain and possible n-connected answers. The study is conducted by implicitly considering the *data complexity* measure (Vardi 1982) (i.e. the complexity where only the ABox is regarded as the input and the other components are assumed to be fixed), which is often considered the more relevant measure when querying KBs as the ABox is typically significantly larger than the other components.

We start by addressing the recognition problems associated with certain and possible n-connected answers, which are parametric w.r.t. a KB language $\mathbb{K}$. We define CERTREC($\mathbb{K}$) (resp. POSSREC($\mathbb{K}$)) as follows: given a consistent KB $\mathcal{K} \in \mathbb{K}$, a UCQ q, and a n-connected answer A, check whether A is certain (resp. possible) for q over $\mathcal{K}$.

We now show that (the data complexity versions of) CERTREC($\mathbb{K}$) and POSSREC($\mathbb{K}$) can be reduced in constant time to the problem of checking whether a suitable Boolean UCQ is entailed by the input KB $\mathcal{K}$ and satisfied by at least one model of the input KB $\mathcal{K}$, respectively.

Proposition 10. *Let q be a UCQ and A be an n-connected answer. It is possible to compute a Boolean UCQ q_A such that, for any consistent KB $\mathcal{K}$, the following holds:*

- *A is certain for q over $\mathcal{K}$ iff $\mathcal{K}$ entails q_A;*
- *A is possible for q over $\mathcal{K}$ iff $\mathcal{I}$ satisfies q_A for at least one $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$.*

Notably, the above result implies that CERTREC($\mathbb{K}$) has exactly the same data complexity as Boolean UCQ entailment over $\mathbb{K}$ KBs, for any $\mathbb{K}$. As for POSSREC($\mathbb{K}$), note that a Boolean UCQ $q = \{() \mid \exists \bar{y}_1.\phi_1(\bar{y}_1) \vee \ldots \vee \exists \bar{y}_m.\phi_m(\bar{y}_m)\}$ is satisfied by at least one model of a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{D} \rangle$ iff at least one of the KBs $\mathcal{K}_{\phi_i} = \langle \mathcal{T}, \mathcal{D} \cup \mathcal{D}_{\phi_i} \rangle$ is consistent, where $\mathcal{D}_{\phi_i}$ is the ABox obtained by freezing ϕ_i.

Importantly, let us consider any KB language $\mathbb{K}$ for which Boolean UCQ entailment (resp. consistency check) is decidable. Then, given any UCQ q and any KB $\mathcal{K} \in \mathbb{K}$, it is possible to construct a set C_n (resp. P_n) of certain (resp. possible) n-answers for q over $\mathcal{K}$ by means of calls to CERTREC($\mathbb{K}$) (resp. POSSREC($\mathbb{K}$)), for any integer n.

Notice, however, that an n-connected answer might be redundant w.r.t. other atoms of C_n (resp. P_n). For example, suppose that $\{\mathsf{ans}_2(a,b)\}$ and $\{\mathsf{ans}_2(b,c)\}$ are certain for q over $\mathcal{K}$, where $a, b, c \in \mathcal{C}$. Despite $A = \{\mathsf{ans}_2(a,x), \mathsf{ans}_2(x,c)\}$ is certain for q over $\mathcal{K}$, where $x \in \mathcal{V}$, in C_n there is a set of atoms of the same cardinality of A that conveys strictly more information than A. Formally, we say that an n-connected answer A is *non-redundant w.r.t. certain (resp. possible) n-answers for q over $\mathcal{K}$* if A is certain (resp. possible) for q over $\mathcal{K}$ and there is no set of atoms $B \subseteq C_n$ (resp. $B \subseteq P_n$) such that (i) $|B| \leq |A|$ and (ii) there is a constant-preserving homomorphism from A to B but not viceversa. In the definition, C_n and P_n are a set of certain and possible n-answers, respectively. Since they are unique up to variable renaming, two sets of atoms that are equal up to variable renaming are considered as the same.

We define NRCERT($\mathbb{K}$) (resp. NRPOSS($\mathbb{K}$)) as the following decision problem: given a consistent KB $\mathcal{K} \in \mathbb{K}$, a UCQ q, and an n-connected answer A, check whether A is non-redundant w.r.t. certain (resp. possible) n-answers for q over $\mathcal{K}$. For these problems, we provide a similar characterization as Proposition 10 does for CERTREC($\mathbb{K}$) and POSSREC($\mathbb{K}$). For NRCERT($\mathbb{K}$) we resort to the query language EQL-Lite(UCQ) and the modal notion of EQL-entailment (Calvanese et al. 2007a; Cima et al. 2023). Intuitively, the expressiveness of the epistemic query language is used to check whether the given n-connected answer A is (i) certain for q over $\mathcal{K}$ and (ii) no *instantiation* of A (obtained by either *equating* two distinct variables or by specializing a variable with an *unknown constant*) is certain for q over $\mathcal{K}$.

Theorem 3. *Let q be a UCQ and A be an n-connected answer. It is possible to compute a Boolean* EQL-Lite(*UCQ*) *query q'_A and two Boolean UCQs q_A and q_B such that, for any consistent KB $\mathcal{K}$, the following holds:*

- *A is non-redundant w.r.t. certain n-answers for q over $\mathcal{K}$ iff $\mathcal{K}$ EQL-entails q'_A;*
- *A is non-redundant w.r.t. possible n-answers for q over $\mathcal{K}$ iff (i) $\mathcal{I}$ satisfies q_A for at least one $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$ and (ii) $\mathcal{I}$ does not satisfy q_B for every $\mathcal{I} \in \mathsf{Mod}(\mathcal{K})$.*

We conclude this section by discussing some relevant concrete cases derivable using the established results. Let $\mathbb{K}$ be any KB language for which both Boolean UCQ entailment and consistency check are in AC_0 in data complexity. Notable KB languages enjoying such property are the DL *DLR-Lite*$_{\mathcal{A},\sqsubseteq}$ (Calvanese et al. 2013) and *acyclic tgds with negative constraints* (Lukasiewicz et al. 2022). By Proposition 10 and the second bullet of Theorem 3, we immediately get that CERTREC($\mathbb{K}$), POSSREC($\mathbb{K}$), and NRPOSS($\mathbb{K}$) are in AC_0 in data complexity. Furthermore, by combining the first bullet of Theorem 3 with (Calvanese et al. 2007a, Theorem 8), we get that NRCERT($\mathbb{K}$) is in AC_0 in data complexity as well. This means that, for such $\mathbb{K}$, dealing with certain/possible n-answers has the same data complexity as standard UCQ entailment. A similar line of reasoning applies for those KB languages $\mathbb{K}$ for which both Boolean UCQ entailment and consistency check are P-complete in data complexity, such as *(frontier-)guarded tgds with negative constraints* (Baget et al. 2011). In these cases, we derive that CERTREC($\mathbb{K}$), POSSREC($\mathbb{K}$), NRCERT($\mathbb{K}$), and NRPOSS($\mathbb{K}$) are all P-complete in data complexity.

7 Conclusion

In this work, we presented a framework for informativeness of query answers in KBs, studied the informativeness of query answers known in the literature, and studied novel forms of answers with their computational characteristics.

A first extension of our results is the definition of answers that preserve the certain or possible knowledge of $\exists$Pos(ANS). Another interesting direction is to design efficient enumeration algorithms for our novel notion of answers and implement them in practice. Finally, our framework could be used to improve the informativeness of answers in different contexts. For example, we foresee applications for ontological queries with aggregation and arithmetic operators or Consistent Query Answering.

Acknowledgements

This work has been supported by MUR under the PNRR project FAIR (PE0000013) and by the EU under the H2020-EU.2.1.1 project TAILOR (grant id. 952215). The authors would like to thank the anonymous referees for their valuable comments and Leonid Libkin for the conversations on the topic of this work.

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

Arenas, M.; Pérez, J.; and Reutter, J. L. 2013. Data exchange beyond complete data. *Journal of the ACM*, 60(4): 28:1–28:59.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press. ISBN 0-521-78176-0.

Baget, J.-F.; Mugnier, M.-L.; Rudolph, S.; and Thomazo, M. 2011. Walking the Complexity Lines for Generalized Guarded Existential Rules. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI 2011)*, 712–717.

Bienvenu, M.; and Ortiz, M. 2015. Ontology-Mediated Query Answering with Data-Tractable Description Logics. In *Reasoning Web. Semantic Technologies for Intelligent Data Access – Eleventh International Summer School Tutorial Lectures (RW 2015)*, volume 9203 of *Lecture Notes in Computer Science*, 218–307.

Borgida, A.; Toman, D.; and Weddell, G. E. 2016. On Referring Expressions in Query Answering over First Order Knowledge Bases. In *Proceedings of the Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2016)*, 319–328.

Borgida, A.; Toman, D.; and Weddell, G. E. 2017. Concerning Referring Expressions in Query Answers. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI 2017)*, 4791–4795.

Calautti, M.; Console, M.; and Pieris, A. 2021. Benchmarking Approximate Consistent Query Answering. In Libkin, L.; Pichler, R.; and Guagliardo, P., eds., *PODS'21: Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*.

Calì, A.; Gottlob, G.; and Kifer, M. 2013. Taming the Infinite Chase: Query Answering under Expressive Relational Constraints. *Journal of Artificial Intelligence Research*, 48: 115–174.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007a. EQL-Lite: Effective First-Order Query Processing in Description Logics. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI 2007)*, 274–279.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007b. Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family. *Journal of Automated Reasoning*, 39(3): 385–429.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2013. Data Complexity of Query Answering in Description Logics. *Artificial Intelligence*, 195: 335–360.

Cima, G.; Console, M.; Lenzerini, M.; and Poggi, A. 2023. Epistemic Disjunctive Datalog for Querying Knowledge Bases. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)*, 6280–6288.

Civili, C.; and Libkin, L. 2018. Approximating Certainty in Querying Data and Metadata. In *Proceedings of the Sixteenth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2018)*, 582–591.

Console, M.; Guagliardo, P.; and Libkin, L. 2016. Approximations and Refinements of Certain Answers via Many-Valued Logics. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR*.

Console, M.; Guagliardo, P.; and Libkin, L. 2022. Propositional and predicate logics of incomplete information. *Artif. Intell.*, 302: 103603.

Console, M.; Guagliardo, P.; Libkin, L.; and Toussaint, E. 2020. Coping with Incomplete Data: Recent Advances. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS*, 33–47. ACM.

Grahne, G.; and Onet, A. 2012. Representation systems for data exchange. In Deutsch, A., ed., *15th International Conference on Database Theory, ICDT '12, Berlin, Germany, March 26-29, 2012*, 208–221. ACM.

Imielinski, T.; and Lipski, W., Jr. 1984. Incomplete Information in Relational Databases. *Journal of the ACM*, 31(4): 761–791.

Levesque, H. J.; and Lakemeyer, G. 2001. *The Logic of Knowledge Bases*. The MIT Press.

Libkin, L. 2016. Certain answers as objects and knowledge. *Artificial Intelligence*, 232: 1–19.

Lipski, W. 1979. On Semantic Issues Connected with Incomplete Information Databases. *ACM Trans. Database Syst.*, 4(3): 262–296.

Lukasiewicz, T.; Malizia, E.; Martinez, M. V.; Molinaro, C.; Pieris, A.; and Simari, G. I. 2022. Inconsistency-tolerant query answering for existential rules. *Artificial Intelligence*, 307: 103685.

Lutz, C.; and Przybylko, M. 2022. Efficiently Enumerating Answers to Ontology-Mediated Queries. In *Proceedings of the Forty-First ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS 2022)*, 277–289.

Lutz, C.; and Przybylko, M. 2023. Efficient Answer Enumeration in Description Logics with Functional Roles. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI 2023)*, 6483–6490.

Vardi, M. Y. 1982. The Complexity of Relational Query Languages (Extended Abstract). In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing (STOC 1982)*, 137–146.