# Answering Counting Aggregate Queries over Ontologies of the *DL-Lite* Family

**Egor V. Kostylev**
University of Edinburgh

**Juan L. Reutter**
PUC Chile and University of Edinburgh

## Abstract

One of the main applications of description logics is the ontology-based data access model, which requires algorithms for query answering over ontologies. In fact, some description logics, like those in the DL-Lite family, are designed so that simple queries, such as conjunctive queries, are efficiently computable. In this paper we study counting aggregate queries over ontologies, i.e. queries which use aggregate functions COUNT and COUNT DISTINCT. We propose an intuitive semantics for certain answers for these queries, which conforms to the open world assumption. We compare our semantics with other approaches that have been proposed in different contexts. We establish data and combined computational complexity for the problems of answering counting aggregate queries over ontologies for several variants of DL-Lite.

## 1 Introduction

The growing popularity of ontologies as a paradigm for representing knowledge in the Semantic Web is based on the ability to describe incomplete information in the domain of interest.

Several variations of the *Web Ontology Language* (*OWL*) have been formalized to manage ontologies. Most of these languages correspond to various decidable fragments of first order logic, which are called *description logics* (*DLs*). However, applications like *ontology-based data access* (*OBDA*) require algorithms not only to decide standard reasoning problems, such as satisfiability and model checking, but also to answer database-style queries (Calvanese et al. 2011; Kontchakov et al. 2011). This motivates the use of description logics of the *DL-Lite* family in, e.g. OWL 2 QL, which have been designed specifically to maximize expressive power while maintaining good query answering properties (Cuenca Grau et al. 2008). In particular, the computational complexity of answering simple queries such as *conjunctive queries* (*CQ*s) and *unions of conjunctive queries* (*UCQs*) over these DLs is the same as for relational databases (Calvanese et al. 2007; Artale et al. 2009).

Some attention has recently been paid to the problem of answering various extensions of CQs and UCQs over on-

tologies. For example (Bienvenu, Ortiz, and Simkus 2012) study path queries over ontologies, while (Rosati 2007) and (Gutiérrez-Basulto, Ibáñez-García, and Kontchakov 2012) consider adding some form of negation to these simple queries. The general conclusion from these papers is that the complexity of evaluation of such queries is usually higher than for CQs and UCQs and even higher than for similar problems in relational databases. In some cases this difference in complexity is surprisingly high: e.g. while answering UCQs with inequalities is known to be efficiently computable for relational database settings, the problem is undecidable when such a query is posed over *DL-Lite* ontologies.

Yet there is another extension of CQs that has received little attention in the context of OBDA — *aggregate queries*. These are the queries that answer questions such as "How many children does Ann have?" or "What is the average salary over each department in the Pandidakterion?" These queries combine various aggregate functions, such as MIN, MAX, SUM, AVERAGE, COUNT and COUNT DISTINCT (Cohen, Nutt, and Sagiv 2007), with a *grouping* functionality, as in the usual GROUP BY clause of relational databases.

Aggregate queries are an important and heavily used part of almost every relational database query language, including SQL. In the context of the Semantic Web we expect a particular need for aggregates in the OBDA settings, with applications such as SPARQL under entailment regimes (Glimm et al. 2013). But despite their importance, the study of aggregate queries over ontologies has been lacking, save for a few exceptions (Calvanese et al. 2008).

The main reason for the lack of research in this direction is the difficulty of defining a semantics for aggregate queries over ontologies. The complication is that, unlike relational databases, in ontologies one assumes that every knowledge base instance is incomplete and describes a part of the infinite number of models of the knowledge base (i.e. the *open world assumption* is assumed), and a query may have a different answer on each of these models. For standard queries like CQs and UCQs this problem is usually overcome by computing the *certain answers* of queries, i.e. the tuples that are answers in all possible models (Calvanese et al. 2007). This approach, however, is not suitable for aggregate queries, as the following shows.

Consider a knowledge base where Ann is a parent and the ontology asserts that every parent has at least one child.

If nothing else is assumed then for every positive integer $n$ there exists a model where Ann has $n$ children. Thus, the answer to a simple query "How many children does Ann have?" in different models of the knowledge base can be any number greater than or equal to $1$. The syntactic intersection of these answers (i.e. applying standard certain answers semantics) trivially gives us the empty set, which is clearly not satisfactory. As a different approach, (Calvanese et al. 2008) introduced *epistemic* semantics for aggregate queries. In a nutshell, the idea is to apply the aggregation function only to known values. For example, the epistemic answer to the query above is $0$, because we do not definitely know anybody who is a child of Ann. But this is clearly not the desired answer: since Ann is a parent we know that she has at least one child. Hence the epistemic semantics does not always give a correct answer when applied to COUNT queries.

As the first contribution of this paper, in Sec. 3 we embark on the task of defining a suitable semantics for answering what we call *counting aggregate queries*, which are queries that use COUNT or COUNT DISTINCT functions. Motivated by the original idea of certain answers, we seek to find the maximal information that is common in the answers to such a query for all the models of a knowledge base. This gives rise to the notion of *aggregate certain answers*, which can be explained as follows: a number is an aggregate certain answer to a counting query over a knowledge base if it does not exceed the result of the query over any model of this knowledge base. For instance, in the above example, even if we do not know precisely how many children Ann has, we know that she has at least one, and thus $1$ is an aggregate certain answer to the query.

Of course this semantics is not well suited for aggregation primitives such as SUM or AVERAGE. But, as we show in this paper, it is a natural and useful semantics for aggregate queries that count.

Having established our semantics, we turn to the study of the algorithmic properties of aggregate certain answers computation for counting queries. We concentrate on ontologies of the *DL-Lite* family, in particular *DL-Lite$_{core}$* and *DL-Lite$_{\mathcal{R}}$* (Calvanese et al. 2007). The choice of these DLs is twofold: first, as mentioned above, these formalisms are important in the OBDA settings; second, they are among the simplest DLs and hence good candidates to begin with.

As usual in the theory of DLs, in Sec. 4 we study these problems assuming that the query and the *terminology* (i.e. the *TBox*) are *fixed*, and the only input is the *assertions* (*ABox*). This corresponds to the *data complexity* of the problem in Vardi's taxonomy (Vardi 1982). Somewhat surprisingly, our results show that the complexity of aggregate certain answers problem is resilient to the choice of both DL and counting function and is coNP-complete in all cases. In order to get a further understanding of the computational properties of the problems, in Sec. 5 we study their *combined complexity*, i.e. assume that the query, ABox and TBox are the input. Here we do find differences: both count distinct and count aggregate query answering are coNExp-Time-complete for *DL-Lite$_{\mathcal{R}}$*; yet the former problem is $\Pi_2^p$-complete and the latter is in coNExpTime for *DL-Lite$_{core}$*. Hereby, the small increase of expressivity from *DL-Lite$_{core}$*

to *DL-Lite$_{\mathcal{R}}$* makes at least the count distinct problem exponentially more difficult. As far as we are aware, these are the first tight complexity bounds for answering aggregate queries in the presence of ontologies.

**Related Work** Although mostly unexplored in the context of ontologies, semantics for aggregate queries have been already defined for other database settings that feature incomplete information. For example, an inconsistent database instance (w.r.t. a set of constraints) describes a set of repairs, each of which satisfies the constraints and can be obtained from the instance by a minimal number of transformations. Aggregate queries over inconsistent databases were explored in (Arenas et al. 2003), where the *range semantics* was defined. Intuitively, this semantics corresponds to the *interval* between the minimal and the maximal possible answers to the query, amongst all the repairs of a given instance. The same semantics was adopted by (Libkin 2006; Afrati and Kolaitis 2008) in the context of data exchange.

However, the techniques from these papers cannot be immediately applied to ontologies, because of several specific properties. In particular, these papers consider variations of the *closed world assumption*, whereas in ontologies the open world assumption is assumed. Furthermore, data exchange settings are based on source-to-target dependencies and weakly acyclic target dependencies. This rules out all types of recursion in ontological knowledge, thus simplifying the study to a great extent.

In the context of ontologies, in (Calvanese et al. 2008) the range semantics itself was claimed to be trivially meaningless for aggregate queries over ontologies. For example, for almost any knowledge base we can construct a model such that the aggregate value of an AVERAGE query evaluates to any number. Similar examples can be given for all other standard aggregate functions, except for COUNT and COUNT DISTINCT, which are precisely the aggregates that are the focus of this paper. As we will show the computation of the upper bound of the range is almost trivial in these cases as well. But the lower bound of the range, i.e. the minimal possible value described above, is completely natural, and by no means trivial to compute. In fact, the lower bound of the range semantics is strongly related to our notion of aggregate certain answers as follows: a number is in the aggregate certain answers if and only if it is less than or equal to the lower bound of the range. Thus, this work on aggregate certain answers can be seen as an adaptation of the range semantics of (Arenas et al. 2003) to ontologies.

## 2 Preliminaries

**Syntax of *DL-Lite*** Let $A_0, A_1, \ldots$ be *atomic concepts* and $P_0, P_1, \ldots$ be *atomic roles*. *Concepts C* and *roles E* of *DL-Lite* languages are formed by the following grammar:

$$
\begin{array}{llll}
B & ::= & A_i \mid \exists R, & R & ::= & P_i \mid P_i^-, \\
C & ::= & B \mid \neg B, & E & ::= & R \mid \neg R.
\end{array}
$$

A *TBox* is a finite set of assertions. In the language of *DL-Lite$_{core}$* the assertions are of the form $B \sqsubseteq C$. In *DL-Lite$_{\mathcal{R}}$* the form $R \sqsubseteq E$ is also allowed.

An *ABox* is a set of assertions of the forms $A_i(a)$ and $P_i(a, b)$ where *constants* $a$, $b$ are from an *active domain* $\mathbb{D}$.

A *knowledge base* (or *KB*) $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ of a *DL-Lite* language contains a TBox $\mathcal{T}$ of the language and an ABox $\mathcal{A}$.
**Semantics of *DL-Lite*** An *interpretation* $\mathcal{I} = (\mathbb{D}^{\mathcal{I}}, \cdot^{\mathcal{I}})$ contains a (possibly infinite) *domain* of *elements* $\mathbb{D}^{\mathcal{I}}$ such that $\mathbb{D} \subseteq \mathbb{D}^{\mathcal{I}}$, and maps each concept $C$ to a subset $C^{\mathcal{I}}$ of $\mathbb{D}^{\mathcal{I}}$ and each role $R$ to a binary relation $R^{\mathcal{I}}$ over $\mathbb{D}^{\mathcal{I}}$ such that

$$(P_i^-)^{\mathcal{I}} = \{(a,b) \mid (b,a) \in P_i^{\mathcal{I}}\}, \quad (\neg B)^{\mathcal{I}} = \mathbb{D}^{\mathcal{I}} \backslash B^{\mathcal{I}},$$
$$(\exists R)^{\mathcal{I}} = \{a \mid \exists b : (a,b) \in R^{\mathcal{I}}\}, \quad (\neg R)^{\mathcal{I}} = \mathbb{D}^{\mathcal{I}} \times \mathbb{D}^{\mathcal{I}} \backslash R^{\mathcal{I}}.$$

An interpretation $\mathcal{I}$ is a *model* of a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ (written $\mathcal{I} \models \mathcal{K}$) if for any assertion $B \sqsubseteq C$ in $\mathcal{T}$ it holds that $B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$, for any $R \sqsubseteq E$ it holds that $R^{\mathcal{I}} \subseteq E^{\mathcal{I}}$, for any $A_i(a)$ in $\mathcal{A}$ it holds that $a \in A_i^{\mathcal{I}}$, and for any $P_i(a,b)$ it holds that $(a,b) \in P_i^{\mathcal{I}}$. In particular, by the definitions above we adopt the *unique name assumption* (*UNA*) on constants which is conventional for *DL-Lite*; dropping this assumption does not affect any result of this paper.
**Conjunctive queries** A *conjunctive query* (or *CQ*) is an expression of the form

$$q(\mathbf{x}) \text{ :- } \exists \mathbf{y} \ \phi(\mathbf{x}, \mathbf{y}), \tag{1}$$

where $\mathbf{x}$ is a tuple of *free* variables, $\mathbf{y}$ is a tuple of *existential* variables, and the *body* $\phi(\mathbf{x}, \mathbf{y})$ is a conjunction of *atoms* of the form $A_i(u)$ or $P_i(u_1, u_2)$, where $u, u_1, u_2$ are variables from $\mathbf{x} \cup \mathbf{y}$.

A CQ (1), holds for an interpretation $\mathcal{I}$ and a tuple $\mathbf{t}$ of elements from $\mathbb{D}^{\mathcal{I}}$ (written $\mathcal{I} \models q(\mathbf{t})$) iff there exists an *evaluation* from $q$ to $\mathbb{D}^{\mathcal{I}}$ for $\mathbf{t}$, i.e. a mapping $h : \mathbf{x} \cup \mathbf{y} \to \mathbb{D}^{\mathcal{I}}$, such that $h(\mathbf{x}) = \mathbf{t}$ and $h(\mathbf{z}) \in S^{\mathcal{I}}$, for every atom $S(\mathbf{z})$ in $\phi(\mathbf{x}, \mathbf{y})$. A tuple $\mathbf{t}$ is in the *certain answer* to a CQ (1) over a KB $\mathcal{K}$ if $\mathcal{I} \models q(\mathbf{t})$ holds for every model $\mathcal{I}$ of $\mathcal{K}$.

## 3 Counting Queries over Ontologies

The ability to evaluate aggregate queries is a default in every DBMS and is in the standard of SQL. However, as mentioned in the introduction, little attention to this type of queries has been paid in the context of ontologies. Starting to fill this gap, in this section we formally define counting aggregate queries over ontologies of *DL-Lite* family and compare this definition with existing notions in related areas.

### 3.1 Syntax and Semantics of Counting Queries

Following e.g. (Cohen, Nutt, and Sagiv 2007), an *aggregate conjunctive query* (or *ACQ*) is an expression of the form

$$q(\mathbf{x}, f(\mathbf{z})) \text{ :- } \exists \mathbf{y} \ \phi(\mathbf{x}, \mathbf{y}, \mathbf{z}), \tag{2}$$

where $\mathbf{x}$ is a tuple of *free* variables, $\mathbf{y}$ is a tuple of *existential* variables and $\mathbf{z}$ is a tuple of *aggregation* variables; the *body* $\phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$ is a conjunction of *atoms* of the form $A_i(u)$ or $P_i(u_1, u_2)$, where $u, u_1, u_2$ are variables from $\mathbf{x} \cup \mathbf{y} \cup \mathbf{z}$; and $f(\mathbf{z})$ is an *aggregation function*. In this paper we consider two such functions: the unary *count distinct* function $Cntd(z)$ and nullary *count* function $Count()$. We refer to these queries as *counting ACQs*.

**Example 1.** Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a knowledge base where $\mathcal{T}$ consists of the assertion *Parent* $\sqsubseteq \exists HasChild$, and $\mathcal{A}$ consists of the assertion *Parent*(Ann). The query

$$q_1(x, Count()) \text{ :- } \exists y \ Parent(x) \wedge HasChild(x, y)$$

is an ACQ using the count function. Intuitively, it is meant to count the children of each parent. The query

$$q_2(Cntd(y)) \text{ :- } \exists x \ Parent(x) \wedge HasChild(x, y)$$

is a count distinct ACQ. This query is meant to count all different children having a parent.

To define the semantics of counting queries over a particular model we again follow (Cohen, Nutt, and Sagiv 2007).

We say that the *core* of an ACQ of the form (2) is the CQ $\bar{q}(\mathbf{x} \cup \mathbf{z}) \text{ :- } \exists \mathbf{y} \ \phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$. Furthermore, let $\mathbb{N}^\infty$ be the set of natural numbers with 0 and $+\infty$.

A *count* ACQ $q(\mathbf{x}, Count())$ holds for an interpretation $\mathcal{I}$, a tuple $\mathbf{t}$ of elements from $\mathbb{D}^{\mathcal{I}}$ and a number $n \in \mathbb{N}^\infty$ (written $\mathcal{I} \models q(\mathbf{t}, n)$) iff $n$ is the number of distinct evaluations from the core $\bar{q}$ to $\mathbb{D}^{\mathcal{I}}$ for $\mathbf{t}$.

A *count distinct* ACQ $q(\mathbf{x}, Cntd(z))$ holds for an interpretation $\mathcal{I}$, a tuple $\mathbf{t}$ of elements from $\mathbb{D}^{\mathcal{I}}$ and a number $n \in \mathbb{N}^\infty$ (written $\mathcal{I} \models q(\mathbf{t}, n)$) iff $n$ is the number of distinct constants $a \in \mathbb{D}^{\mathcal{I}}$ such that $\mathcal{I} \models \bar{q}(\mathbf{t}, a)$ for the core $\bar{q}$ of $q$.

**Example 2.** Coming back to Ex. 1, consider the interpretation $\mathcal{I}$ where $Parent^{\mathcal{I}} = \{\text{Ann}\}$ and $HasChild^{\mathcal{I}} = \{(\text{Ann}, \text{Joe})\}$, which is clearly a model for $\mathcal{K}$. Then it is not difficult to see that $\mathcal{I} \models q_1(\text{Ann}, 1)$ and $\mathcal{I} \models q_2(1)$.

For the model $\mathcal{J}$ such that $Parent^{\mathcal{J}} = \{\text{Ann}, \text{Peter}\}$ and $HasChild^{\mathcal{J}} = \{(\text{Ann}, \text{Joe}), (\text{Ann}, \text{Rose}), (\text{Peter}, \text{Joe})\}$, it holds that $\mathcal{J} \models q_1(\text{Ann}, 2)$, $\mathcal{J} \models q_1(\text{Peter}, 1)$ and $\mathcal{J} \models q_2(2)$.

### 3.2 Certain Answers of Counting Queries over Ontologies

A knowledge base normally describes not a single model, but a part of the infinite number of them. That is why one is usually interested in computing the *certain answers* of queries over a KB, i.e. the answers that hold in every model of the KB. As in Sec. 2 the certain answer for a CQ over a KB is just the intersection of the answers to the CQ over all models. In fact, a similar approach is adopted for many other studied query formalisms, such as unions of CQs (Calvanese et al. 2007) and CQs with inequalities (Rosati 2007).

However, in the case of ACQs the definition of certain answers based on such a syntactical intersection is of little use, since it would almost always be empty. For instance, for the query $q_1$ from Ex. 1 and 2 we have that $\mathcal{I} \models q_1(\text{Ann}, 1)$, and $\mathcal{I} \not\models q_1(\text{Ann}, 2)$, yet $\mathcal{J} \not\models q_1(\text{Ann}, 1)$ and $\mathcal{J} \models q_1(\text{Ann}, 2)$. This suggests avoiding using such a syntactic intersection when defining the semantics of ACQs over ontologies.

In the context of OBDA this problem has been identified before by (Calvanese et al. 2008). Their solution was to concentrate only on aggregating over *epistemic* knowledge, i.e. over values which are explicitly mentioned in the ABox of a KB. Such epistemic aggregate queries usually have a non-empty certain answer, based on the intersection, for all standard aggregate queries, including $Max$ and $Average$. However, for counting queries this answer may be somehow non-satisfactory. For example, the epistemic answer to the ACQ $q_1$ over $\mathcal{K}$ from Ex. 1 is $(\text{Ann}, 0)$, because we do not know anybody who is definitely a child of Ann.

That is why we suggest the following definition of certain answers of counting ACQs over DLs, which is essentially

the *minimum* over possible values of the counting function over all the models of a KB. In particular, our certain answer to the query $q_1$ over $\mathcal{K}$ from Ex. 1 contains (Ann, 1), which reflects the fact that we definitely know that Ann has at least one child in any model. We deem this definition to be in line with the open world assumption, adopted in ontologies.

**Definition 3.** *A number* $n \in \mathbb{N}^\infty$ *is in the* aggregate certain answers $Cert(q, \mathbf{t}, \mathcal{K})$ *for a counting ACQ* $q$, *tuple of elements* $\mathbf{t}$, *and a KB* $\mathcal{K}$ *iff* $n \leq \min_{\mathcal{I} \models \mathcal{K}} \{k \mid \mathcal{I} \models q(\mathbf{t}, k)\}$.

Note that a definition like above is non-trivial only for counting standard aggregate queries. Indeed, it relies on their simple property that the minimum above can potentially be any number greater than or equal to 0. For other aggregation functions it is not the case: e.g. such a minimum for $Average$ is trivially almost always $-\infty$.

### 3.3 Range Semantics of Aggregate Queries

As mentioned in the introduction, aggregate queries have been explored in other settings. For example, in the context of inconsistent databases (Arenas et al. 2003) the *range semantics* of aggregates was defined (it was later adopted in data exchange (Libkin 2006; Afrati and Kolaitis 2008)). This semantics focuses on the *interval* of possible aggregation values over all models. In the context of counting ACQs over ontologies it can be defined as follows.

The *range of answers* for a counting ACQ $q$, a tuple $\mathbf{t}$, and a KB $\mathcal{K}$ is the interval $[m(q, \mathbf{t}, \mathcal{K}), M(q, \mathbf{t}, \mathcal{K})]$, where

$$m(q, \mathbf{t}, \mathcal{K}) = \min_{\mathcal{I} \models \mathcal{K}} \{k \mid \mathcal{I} \models q(\mathbf{t}, k)\},$$

$$M(q, \mathbf{t}, \mathcal{K}) = \max_{\mathcal{I} \models \mathcal{K}} \{k \mid \mathcal{I} \models q(\mathbf{t}, k)\}.$$

It is easy to see that the lower bound of the range interval coincides with the maximal certain answer from Def. 3. Considering the upper bound, let's come back to Ex. 1. We can find a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \models q_1(\text{Ann}, n)$ for any number $n \geq 1$, i.e. in this case the upper bound is $+\infty$. The following proposition says that this is not an unusual case.

**Proposition 4.** *Given a counting ACQ* $q$, *a tuple of elements* $\mathbf{t}$, *and a DL-Lite KB* $\mathcal{K}$ *the value* $M(q, \mathbf{t}, \mathcal{K})$ *belongs to the set* $\{0, 1, +\infty\}$, *and can be computed in polynomial time (in the size of* $q$ *and* $\mathcal{K}$).

*Proof (sketch).* Indeed, $M(q, \mathbf{t}, \mathcal{K}) = 0$ iff $\langle \mathcal{T}, \mathcal{A} \cup \mathcal{A}_q \rangle$ has no model, where $\mathcal{A}_q$ is an ABox over the variables of $q$ as constants containing the atoms of $q$ as assertions. Otherwise, we have that $M(q, \mathbf{t}, \mathcal{K}) = 1$ only if $q$ uses $count()$ and has no existentially quantified variables. In all the remaining cases we have that $M(q, \mathbf{t}, \mathcal{K}) = +\infty$, since nothing prevents a model with an infinite number of witnesses. □

Based on this proposition, we may say that the aggregate certain answers semantics from Def. 3 is just an adaptation of the range semantics of (Arenas et al. 2003) to ontologies.

## 4 Data Complexity of Counting Queries

It has been argued many times that in usual database settings the size of the query and the TBox is much smaller than the size of the ABox (see e.g. (Vardi 1982) as a more general statement and (Calvanese et al. 2007) in the context of DL's). This is why in query answering over ontologies one usually explores data complexity of problems, i.e. only database knowledge from ABox is considered as part of the input. In this section we do the same for aggregate certain answers. Formally, let $\mathcal{X} \in \{core, \mathcal{R}\}$, $\mathcal{T}$ be a TBox over *DL-Lite*$_\mathcal{X}$ and $q(\mathbf{x}, f(z))$ be a counting ACQ. We are interested in the following family of problems:

---

*DL-Lite*$_\mathcal{X}$ $f$-AGGREGATE CERTAIN ANSWERS$(\mathcal{T}, q)$
   **Input:**     ABox $\mathcal{A}$, tuple $\mathbf{t}$, and number $n \in \mathbb{N}^\infty$.
   **Question:**   Is $n \in Cert(q, \mathbf{t}, \langle \mathcal{T}, \mathcal{A} \rangle)$?

---

### 4.1 Count Queries
We start with the lower bound for count ACQs.

**Lemma 5.** *There exist a DL-Lite$_{core}$ TBox* $\mathcal{T}$ *and a count ACQ* $q$ *without free variables such that checking whether* $n \in Cert(q, \mathbf{t}_\emptyset, \mathcal{K})$, *where* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, *for an ABox* $\mathcal{A}$, *a number* $n$, *and the empty tuple* $\mathbf{t}_\emptyset$ *is* coNP-*hard*.

*Proof (sketch).* Let $A, B$ and $E, P$ be atomic concepts and roles. Let $q(Count())$ :- $\exists y_1 \ldots y_4\, B(y_1) \wedge E(y_2, y_3) \wedge P(y_2, y_4) \wedge P(y_3, y_4)$ and $\mathcal{T} = \{A \sqsubseteq \exists P, \exists P^- \sqsubseteq B\}$.

Consider the complement of the NP-complete 3-colouring problem with an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ as input and positive output iff the graph has no 3-colouring.

Let $\mathbb{D} = \mathcal{V} \cup \{r, g, b, a\}$. Let $\mathcal{A}$ contain $E(u, v)$ and $E(v, u)$ for each $(u, v) \in \mathcal{E}$, $A(v)$ for each $v \in \mathcal{V}$, $B(c)$ for each $c \in \{r, g, b\}$, and $E(a, a), P(a, r)$.

It holds that $4 \in Cert(q, \mathbf{t}_\emptyset, \mathcal{K})$ iff $\mathcal{G}$ has no 3-colouring. □

Thus, the data complexity of count queries rises from P in the standard database case at least to coNP for *DL-Lite* knowledge bases. The following lemma establishes a matching upper bound for the problem.

**Lemma 6.** *Let* $\mathcal{T}$ *be a fixed DL-Lite$_\mathcal{R}$ TBox and* $q(\mathbf{x}, Count())$ *be a fixed count ACQ. Checking whether* $n \in Cert(q, \mathbf{t}, \mathcal{K})$, *where* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, *for an ABox* $\mathcal{A}$, *a tuple* $\mathbf{t}$, *and a number* $n$ *can be done in* coNP.

*Proof (sketch).* Given an interpretation $\mathcal{J}$ and a number $k$, it is well known that checking whether $\mathcal{J} \models \mathcal{K}$ and $\mathcal{J} \models q(\mathbf{t}, k)$ is in polynomial time (since $q$ is fixed). Hence, it is enough to prove that if there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \models q(\mathbf{t}, n_0)$ for a number $n_0$ then there exists a model $\bar{\mathcal{I}}$ of $\mathcal{K}$ of polynomial size in the size of $\mathcal{A}$ such that $\bar{\mathcal{I}} \models q(\mathbf{t}, \bar{n})$ for some number $\bar{n} \leq n_0$.

Note that $\mathcal{K}$ always has a model with a domain no bigger than $|\mathbb{D}| + |\mathcal{T}|$, so w.l.o.g. we may assume that $n_0 \leq (|\mathbb{D}| + |\mathcal{T}|)^{|q|}$ (which is polynomial since $q$ is fixed).

Fix $\mathcal{I}$ as above. There exists a homomorphism $f : Can(\mathcal{K}) \to \mathcal{I}$, where $Can(\mathcal{K})$ is the *canonical model* of $\mathcal{K}$ (see the definition in e.g. (Calvanese et al. 2007)). W.l.o.g. we assume that it is surjective, i.e. $f(Can(\mathcal{K})) = \mathcal{I}$; since otherwise we could drop elements and assertions of $\mathcal{I}$ which are not in the image of $f$, without increasing $n_0$.

Let $\mathbb{D}^*$ be all elements of $\mathbb{D}^\mathcal{I}$ which are either constants from $\mathbb{D}$ or images of variables by homomorphisms from the

body of $q$ to $\mathcal{I}$. We can construct an interpretation $\hat{\mathcal{I}}$ with the domain $\mathbb{D}^{\hat{\mathcal{I}}} = \cup_{d \in \mathbb{D}^{\mathcal{I}} \setminus \mathbb{D}^*} f^{-1}(d) \cup \mathbb{D}^*$ and with a surjective homomorphism from $Can(\mathcal{K})$ so that $\hat{\mathcal{I}} \models \mathcal{K}$ and $\hat{\mathcal{I}} \models q(\mathbf{t}, \bar{n})$ for some $\bar{n} \leq n_0$.

For every element $d \in \mathbb{D}^{\hat{\mathcal{I}}} \setminus \mathbb{D}^*$ define $\mathcal{N}_q(d)$ as a subinterpretation of $\mathbb{D}^{\hat{\mathcal{I}}}$ induced by all elements reachable from $d$ by an (undirected) path though roles of length no more than $|q|$ and without intermediate nodes from $\mathbb{D}^*$. Define equivalence $\mathcal{N}_q(d) \sim \mathcal{N}_q(d')$ if there exists an isomorphism between $\mathcal{N}_q(d)$ and $\mathcal{N}_q(d')$ preserving $\mathbb{D}^*$.

Note that every element of the canonical model which is not in $\mathbb{D}$, has at most $|\mathcal{T}| + 1$ immediate neighbours. Hence each $d \in \mathbb{D}^{\hat{\mathcal{I}}} \setminus \mathbb{D}^*$ also has at most $|\mathcal{T}| + 1$ immediate neighbours in $\hat{\mathcal{I}}$. Moreover, it holds that $|\mathbb{D}^*| \leq n_0|q| + |\mathbb{D}|$. So, each $\mathcal{N}_q(d)$ is of polynomial size and there is only a polynomial number of equivalence classes induced by $\sim$. Hence, the model $\bar{\mathcal{I}}$ obtained from $\hat{\mathcal{I}}$ by merging all $d_1, d_2$ such that $\mathcal{N}_q(d_1) \sim \mathcal{N}_q(d_2)$ is as required, since such merging does not create new homomorphisms of the body of $q$. $\square$

Note that the lower bound was shown for *DL-Lite$_{core}$*, while the upper bound holds for any *DL-Lite$_{\mathcal{R}}$* KB. Since *DL-Lite$_{\mathcal{R}}$* is more expressive than *DL-Lite$_{core}$*, the lemmas above give us the following complexity result.

**Theorem 7.** *The problem DL-Lite$_{\mathcal{X}}$ Count-*AGGREGATE CERTAIN ANSWERS$(\mathcal{T}, q)$ *is* coNP-*complete in data complexity for any* $\mathcal{X} \in \{core, \mathcal{R}\}$.

## 4.2 Count Distinct Queries

As promised, the coNP complexity bounds also apply for count distinct queries. We start with the lower bound.

**Lemma 8.** *There exist a DL-Lite$_{core}$ TBox $\mathcal{T}$ of one rule and a count distinct ACQ $q$ without free variables such that checking whether $n \in Cert(q, \mathbf{t}_\emptyset, \mathcal{K})$, where $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, for an ABox $\mathcal{A}$ and a number $n$ is* coNP-*hard.*

*Proof (sketch).* Consider $q(Cntd(z)) :\text{-} \exists y_1 \dots y_4 \, P(y_1, z) \wedge R(y_1, y_2) \wedge P(y_2, y_3) \wedge P(y_4, y_3) \wedge E(y_4, y_2)$ and a TBox $\mathcal{T} = \{\exists E \sqsubseteq \exists P\}$, where $E, P$ and $R$ are atomic roles.

Consider the complement of the 3-colouring problem with the input graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ as in the proof of Lem. 5.

Let $\mathbb{D}$ contain the set of elements $\{v, v_1, v_2, v_3, v_4, v_5\}$ for each $v \in \mathcal{V}$. Let $\mathcal{A}$ contain the assertions $E(u, v)$ and $E(v, u)$ for each $(u, v) \in \mathcal{E}$ and the assertions $R(v, v_1), P(v_1, v_2), P(v_3, v_2), E(v_3, v_1), R(v_4, v), P(v_4, v_5)$ for each $v \in \mathcal{V}$.

It holds that $4 \in Cert(q, \mathbf{t}_\emptyset, \mathcal{K})$ iff $\mathcal{G}$ has no 3-colouring. $\square$

The matching algorithm is similar to the count case.

**Lemma 9.** *Let $\mathcal{T}$ be a fixed DL-Lite$_{\mathcal{R}}$ TBox and $q(\mathbf{x}, Cntd(z))$ be a fixed count distinct ACQ. Checking whether $n \in Cert(q, \mathbf{t}, \langle \mathcal{T}, \mathcal{A} \rangle)$ for an ABox $\mathcal{A}$, a tuple $\mathbf{t}$, and a number $n$ can be done in* coNP.

*Proof (sketch).* The proof goes the same lines as the proof of Lem. 6 except that we may bound $n_0$ by $|\mathbb{D}| + |\mathcal{T}|$, and include into $\mathbb{D}^*$ the active domain $\mathbb{D}$ and all homomorphic images of the aggregation variable $z$ to $\mathcal{I}$. $\square$

The lemmas above give a similar result as Thm. 7.

**Theorem 10.** *The problem DL-Lite$_{\mathcal{X}}$ Cntd-*AGGREGATE CERTAIN ANSWERS$(\mathcal{T}, q)$ *is* coNP-*complete in data complexity for any* $\mathcal{X} \in \{core, \mathcal{R}\}$.

# 5 Combined Complexity of Counting Queries

As pointed out in Sec. 4 data complexity is the most used measure of algorithms in any database settings. However, combined complexity has its own value for understanding fundamental properies of problems. In this section we study the combined complexity of computing aggregate certain answers. Formally, let $\mathcal{X} \in \{core, \mathcal{R}\}$ and $f$ be a counting aggregate function. Now we are interested in the following family of problems:

| *DL-Lite$_{\mathcal{X}}$* $f$-AGGREGATE CERTAIN ANSWERS |
|---|
| **Input:** KB $\mathcal{K}$ over *DL-Lite$_{\mathcal{X}}$*, $f$ query $q$, tuple $\mathbf{t}$, and number $n \in \mathbb{N}^\infty$. |
| **Question:** Is $n \in Cert(q, \mathbf{t}, \mathcal{K})$? |

## 5.1 Count Queries

We start again with count queries. Recall the algorithm to compute the certain answers for count queries explained in the proof of Lem. 6. Note that, if one takes into consideration the size of the query and the TBox, then this algorithm naturally gives a coNExpTime upper bound; the only difference is that in this case the number of neighbourhoods is of exponential size (w.r.t. $q$ and $\mathcal{T}$), and thus the instance we need to guess is of exponential size. Next we show that this bound is tight for *DL-Lite$_{\mathcal{R}}$*.

**Lemma 11.** *The problem DL-Lite$_{\mathcal{R}}$ Count-*AGGREGATE CERTAIN ANSWERS *is* coNExpTime-*hard.*

*Proof (idea).* The proof is by a reduction from the complement of the satisfiability problem for first-order logic (FO) formulas in the Bernays-Schöfinkel class (Börger, Grädel, and Gurevich 2001). This class contains all FO formulae of form $\exists \mathbf{x} \, \forall \mathbf{y} \, \psi(\mathbf{x}, \mathbf{y})$, with $\psi$ a quantifier-free formula not using function symbols or equalities. The reduction is inspired by the techniques used in (Arenas, Barceló, and Reutter 2011) to show coNExpTime-hardness of query answering problems in data exchange context. The idea is as follows. It is known that a formula in the form above has a model iff it has a model using at most $|\mathbf{x}|$ elements. We can show how to construct a KB $\mathcal{K}$ in which each model $\mathcal{I}$ of $\mathcal{K}$ represents a model over the vocabulary of the formula (of size exponential in $\mathbf{x}$ and $\psi$), and a query $q$ such that the formula is satisfiable iff there is a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \models q(\mathbf{t}_\emptyset, 2)$ for the empty tuple $\mathbf{t}_\emptyset$. Then 3 is in the aggregate certain answers iff the formula is not satisfiable. $\square$

Unfortunately, the reduction above uses role inclusions in the TBox, i.e. it is applicable only to *DL-Lite$_{\mathcal{R}}$*. We leave open the exact complexity of the *DL-Lite$_{core}$* Count-ACQ ANSWERING problem, although it is not difficult to adapt the results of the following section to obtain a $\Pi_2^p$ lower bound. The following theorem summarizes our results.

**Theorem 12.** *(1) The problem DL-Lite$_{core}$ Count-*AGGREGATE CERTAIN ANSWERS *is in* coNExpTime. *(2) The problem DL-Lite$_{\mathcal{R}}$ Count-*AGGREGATE CERTAIN ANSWERS *is* coNExpTime-*complete.*

## 5.2 Count Distinct Queries

Just as we did for count queries, we can easily obtain a coNExpTime upper bound for count distinct ones from the proof of Lem. 9. However, in this case we can do much better if we restrict ourselves to *DL-Lite$_{core}$*.

**Lemma 13.** *There exists a $\Pi_2^p$-algorithm which solves the DL-Lite$_{core}$ Cntd-*AGGREGATE CERTAIN ANSWERS *problem.*

*Proof (sketch).* The combined complexity of the algorithm from Lem. 9 is exponential, since sub-interpretations $\mathcal{N}_q(d)$ can be of exponential size. Next we show how to redefine these sub-interpretations to have them polynomial (for *DL-Lite$_{core}$*) but still keep the possibility of merging them without increasing the number $\bar{n}$.

For every pair of variables $u, v$ from the body $\phi(\mathbf{x}, \mathbf{y}, z)$ of $q$ let $\mathcal{L}_q(u, v)$ be the sub-interpretation of $\phi(\mathbf{x}, \mathbf{y}, z)$ induced by all variables $w$ on simple paths from $u$ to $v$.

For every $d \in \mathbb{D}^{\hat{\mathcal{I}}} \backslash \mathbb{D}^*$ (where $\hat{\mathcal{I}}$ and $\mathbb{D}^*$ are as in the proof of Lem. 9) define $\mathcal{N}_q^*(d)$ as a sub-interpretation of $\mathbb{D}^{\hat{\mathcal{I}}}$ induced by all elements $d'$ such that there exists $u, v \in \mathbf{x} \cup \mathbf{y} \cup z$ and a homomorphism $h$ from $\mathcal{L}_q(u, v)$ to $\mathbb{D}^{\hat{\mathcal{I}}}$ such that $h(u) = d$, $h(v) = d'$ and $h(w) \notin \mathbb{D}^*$ for all $w \neq v$.

Since every $d_1 \in \mathbb{D}^{\hat{\mathcal{I}}} \backslash \mathbb{D}^*$ and every $R$ have at most one $d_2$ such that $\hat{\mathcal{I}} \models R(d_1, d_2)$, every pair $u, v$ induces at most one element in every $\mathcal{N}_q^*(d)$.[1] Hence $\mathcal{N}_q^*(d)$ is of polynomial size. However, merging $d_1$ and $d_2$ in $\hat{\mathcal{I}}$ such that $\mathcal{N}_q^*(d_1) \sim \mathcal{N}_q^*(d_2)$, does not create new homomorphic images of $z$, so it does not increase $\bar{n}$. $\square$

In this case we have the matching lower bound.

**Lemma 14.** *The problem DL-Lite$_{core}$ Cntd-*AGGREGATE CERTAIN ANSWERS *is* $\Pi_2^p$-*hard.*

*Proof (sketch).* Consider the $\Pi_2^p$-complete $\forall\exists$ 3-SAT problem whose input is a 3-CNF Boolean formula $\psi = \forall x_1, \ldots, x_n \exists z_1, \ldots, z_m \bigwedge_{1 \leq k \leq \ell} \psi_k$, where each $\psi_k$ contains exactly 3 literals (over variables denoted $y_k^1, y_k^2, y_k^3$).

Consider the Boolean *Cntd*-ACQ $q(Cntd(u))$ :- $\exists \mathbf{w}$ $V(s, u) \wedge \bigwedge_{k=1}^{\ell}(R(s, c_k) \wedge S_1(c_k, v_{y_k^1}) \wedge S_2(c_k, v_{y_k^2}) \wedge S_3(c_k, v_{y_k^3}) \wedge C_k^1(y_k^1) \wedge C_k^2(y_k^2) \wedge C_k^3(y_k^3)) \wedge \bigwedge_{i=1}^{n} V(x_i, v_{x_i}) \wedge \bigwedge_{j=1}^{m} V(z_j, v_{z_j})$, with corresponding roles and concepts, where $\mathbf{w}$ is the tuple of all the variables above, except $u$.

Let $\mathcal{A}$ be an ABox (over the constants below) containing:
(a) a copy of the body of $q$, except $V(s, u)$, such that every variable $a$ (except $u$) is "frozen" into the constant $\hat{a}$;
(b) $R(\bar{x}_i, \hat{c}_k)$ and $X_i(\bar{x}_i)$ for each $1 \leq i \leq n$ and $1 \leq k \leq \ell$;
(c) $V(\hat{s}, 0), V(\hat{s}, 1), V(\bar{z}_j, 0), V(\bar{z}_j, 1)$ for each $1 \leq j \leq m$;

| DL-Lite | Data complexity | | Combined complexity | |
|---|---|---|---|---|
| | Count | Cntd | Count | Cntd |
| *core* | coNP-c | coNP-c | in coNExp | $\Pi_2^p$-c |
| $\mathcal{R}$ | coNP-c | coNP-c | coNExp-c | coNExp-c |

Table 1: A summary of the complexity results. Here "-c" stands for "-complete" and coNExp – for coNExpTime.

(d) $C_k^1(\bar{y}_k^1), C_k^2(\bar{y}_k^2), C_k^3(\bar{y}_k^3)$ for each $1 \leq k \leq \ell$, where $\bar{y}_k^1, \bar{y}_k^2, \bar{y}_k^3 \in \{\bar{x}_1, \ldots, \bar{x}_n, \bar{z}_1, \ldots, \bar{z}_m\}$ correspond to $y_k^1, y_k^2, y_k^3$;
(e) $S_1(\bar{c}_k^p, \sigma_p(y_k^1)), S_2(\bar{c}_k^p, \sigma_p(y_k^2))$ and $S_3(\bar{c}_k^p, \sigma_p(y_k^3))$ for each $1 \leq k \leq \ell$ and each satisfying assignment $\sigma_p$, $1 \leq p \leq 7$, of $\psi_k$ (evaluating to one of the constants 0 or 1);
(g) $V(d_1, d_2)$ and $R(d_1, \bar{c}_k^p)$ for each $1 \leq k \leq \ell$, $1 \leq p \leq 7$.

Finally, the TBox $\mathcal{T}$ contains $X_i \sqsubseteq \exists V$ for all $1 \leq i \leq n$.

We have that $3 \in Cert(q, \mathbf{t}_{\emptyset}, \langle \mathcal{T}, \mathcal{A} \rangle)$ iff for all valuations for $x_1, \ldots, x_n$ there is a valuation for $z_1, \ldots, z_m$ such that $\psi$ holds. $\square$

The only remaining question is whether the algorithm for computing aggregate certain answers over *DL-Lite$_{\mathcal{R}}$* knowledge bases is optimal. We settle this with our last lemma.

**Lemma 15.** *The problem DL-Lite$_{\mathcal{R}}$ Cntd-*AGGREGATE CERTAIN ANSWERS *is* coNExpTime-*hard.*

*Proof (idea).* The proof is again by a reduction from satisfiability of FO formulas in the Bernays-Schöfinkel class. The reduction is similar to the one explained for count queries in the proof of Lem. 11, albeit considerably more technical, since this time one needs to take extra care when defining the query, so that only the correct parts of this query are always mapped to a model. We have given an example of how to correctly apply this technique in the proof of Lem. 14. $\square$

Summing up, we have our last theorem.

**Theorem 16.** *(1) The problem DL-Lite$_{core}$ Cntd-*AGGREGATE CERTAIN ANSWERS *is* $\Pi_2^p$-*complete. (2) The problem DL-Lite$_{\mathcal{R}}$ Cntd-*AGGREGATE CERTAIN ANSWERS *is* coNExpTime-*complete.*

## 6 Conclusion

In this paper we have defined an intuitive semantics for counting aggregate queries over ontologies and explored the computational complexity of the corresponding problems. The results, summarized in Table 1, show that the problems are decidable, but intractable. Hence, heuristics and approximations for answering ACQs are on high demand from the practical point of view, with applications, for instance, in the definition of general aggregation in SPARQL under entailment regimes. We consider the epistemic semantics as one of such approximations, since it has lower data complexity but does not always provide the desired answer. Our work settles the theoretical foundations for further discussion.

---

[1] This is the argument which is not valid for *DL-Lite$_{\mathcal{R}}$*.

# References

Afrati, F., and Kolaitis, P. G. 2008. Answering aggregate queries in data exchange. In *Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '08, 129–138. New York, NY, USA: ACM.

Arenas, M.; Barceló, P.; and Reutter, J. L. 2011. Query languages for data exchange: Beyond unions of conjunctive queries. *Theory Comput. Syst.* 49(2):489–564.

Arenas, M.; Bertossi, L.; Chomicki, J.; He, X.; Raghavan, V.; and Spinrad, J. 2003. Scalar aggregation in inconsistent databases. *Theor. Comput. Sci.* 296(3):405–434.

Artale, A.; Calvanese, D.; Kontchakov, R.; and Zakharyaschev, M. 2009. The DL-Lite family and relations. *J. Artif. Intell. Res. (JAIR)* 36:1–69.

Bienvenu, M.; Ortiz, M.; and Simkus, M. 2012. Answering expressive path queries over lightweight DL knowledge bases. In Kazakov, Y.; Lembo, D.; and Wolter, F., eds., *Description Logics*, volume 846 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Börger, E.; Grädel, E.; and Gurevich, Y. 2001. *The Classical Decision Problem*. Springer, Berlin.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. of Automated Reasoning* 39(3):385–429.

Calvanese, D.; Kharlamov, E.; Nutt, W.; and Thorne, C. 2008. Aggregate queries over ontologies. In Elmasri, R.; Doerr, M.; Brochhausen, M.; and Han, H., eds., *ONISW*, 97–104. ACM.

Calvanese, D.; Giacomo, G. D.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rodriguez-Muro, M.; Rosati, R.; Ruzzi, M.; and Savo, D. F. 2011. The MASTRO system for ontology-based data access. *Semantic Web* 2(1):43–53.

Cohen, S.; Nutt, W.; and Sagiv, Y. 2007. Deciding equivalences among conjunctive aggregate queries. *Journal of the ACM* 54(2).

Cuenca Grau, B.; Horrocks, I.; Motik, B.; Parsia, B.; Patel-Schneider, P.; and Sattler, U. 2008. Owl 2: The next step for OWL. *Web Semant.* 6(4):309–322.

Glimm, B.; Ogbuji, C.; Hawke, S.; Herman, I.; Parsia, B.; Polleres, A.; and Seaborne, A. 2013. SPARQL 1.1 entailment regimes. W3C Recommendation 21 March 2013, `http://www.w3.org/TR/2013/REC-sparql11-entailment-20130321/`.

Gutiérrez-Basulto, V.; Ibáñez-García, Y. A.; and Kontchakov, R. 2012. An update on query answering with restricted forms of negation. In *Proceedings of the 6th international conference on Web Reasoning and Rule Systems*, RR'12, 75–89. Berlin, Heidelberg: Springer-Verlag.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2011. The combined approach to ontology-based data access. In *IJCAI*, 2656–2661.

Libkin, L. 2006. Data exchange and incomplete information. In Vansummeren, S., ed., *PODS*, 60–69. ACM.

Rosati, R. 2007. The limits of querying ontologies. In Schwentick, T., and Suciu, D., eds., *ICDT*, volume 4353 of *Lecture Notes in Computer Science*, 164–178. Springer.

Vardi, M. Y. 1982. The complexity of relational query languages (extended abstract). In *STOC*, 137–146.