Controlled Query Evaluation over Lightweight Ontologies*

B. Cuenca Grau, E. Kharlamov, E. V. Kostylev, and D. Zheleznyakov

Department of Computer Science, University of Oxford

Abstract. We study confidentiality enforcement in ontologies under the Controlled Query Evaluation (CQE) framework. In a CQE system, a *policy* specifies the sensitive information, and a *censor* ensures that answers to user's queries that could violate the policy are not returned. Our goal is the design of *optimal* CQE algorithms, which ensure confidentiality while maximising access to information. We study two natural classes of censors that can be realised using existing infrastructure for query answering and propose optimal CQE algorithms for the standardised profiles of the ontology language OWL 2.

1 Introduction

As ontology-based information systems are becoming increasingly mature, there is a pressing need to devise mechanisms for ensuring that data is only made accessible to authorised users [1, 7–12, 20, 21].

Controlled Query Evaluation (CQE) is a prominent formal framework for confidentiality enforcement. In CQE, sensitive information is declaratively specified by means of a *policy* and confidentiality is enforced by a *censor*: when given a user query, a censor checks whether returning the answer might lead to a policy violation, in which case it returns a distorted answer. CQE was introduced in [18], and studied in [3, 4, 6] for propositional databases with complete information. It was extended to (propositional) incomplete databases in [5]. Beyond propositional logic, CQE remains largely unexplored [2].

In this paper we study CQE in the context of ontologies. Our basic framework is described in Section 3. We assume data to be hidden and that users interact with the system by means of a query interface. An ontology, which we assume to be known to all users, provides the vocabulary and background knowledge needed for users to formulate accurate queries, as well as to enrich query answers with implicit information. Policies are given as a set of ground atoms that follow from the ontology and data. When given a (conjunctive) query, the system returns the subset of certain answers determined by the censor; in this way, the role of the censor is to preserve confidentiality by filtering out those answers that could lead to a violation of the policy. Formally, we model the information that users could gather by querying the system as an (infinite) first-order theory;

* This paper recapitulates and extends our previous work [11]. It comes with a technical report available at http://tinyurl.com/DL14paper55.

confidentiality preservation then amounts to ensuring that such theory together with the ontology does not entail any atom in the policy. In this setting, there is a danger that confidentiality enforcement may over-restrict users' access. Thus, we focus on *optimal* censors, which maximise answers to queries while ensuring confidentiality of the policy. Furthermore, we are interested in censors that can be implemented by reusing off-the-shelf query answering infrastructure. To fulfil this requirement, we introduce in Section 4 two classes of censors, which we call *view* and *obstruction* censors, respectively.

View censors return only those answers that follow from the ontology and a materialised dataset (a *view*). In this way, a view encodes the information that users are authorised to access: the censor answers faithfully all queries against the view, and any information not captured by the view is inaccessible by default. View censors require the ability to materialise implicit data, and hence are especially well-suited for RDF-based applications in which reasoning is performed by a triple store. Obstruction censors are dual to view censors in the sense that they explicitly specify information which users are denied access to (with all other information being accessible by default). Obstruction censors are specified by a finite set of "forbidden query patterns" (*obstructions*), and all query answers that instantiate such patterns are filtered out. In contrast to view censors, obstruction censors do not require modification of the data and hence are well-suited for OBDA applications, where data is typically managed by an RDBMS. We finally characterise the duality of views and obstructions and argue that it is not always possible to "simulate" one with another.

In Section 5 we focus on view censors. First, we investigate their intrinsic limitations, and then show how these limitations can be circumvented. We propose algorithms for computing optimal view censors for knowledge bases with OWL 2 RL, EL and QL ontologies under relatively minor restrictions. Our algorithms, however, rely on views that can be of exponential size in the worst case. So, we identify natural conditions on ontologies that guarantee polynomial size of optimal views. In particular, all OWL 2 QL ontologies satisfy these conditions.

In Section 6 we turn our attention to obstruction censors and provide sufficient and necessary conditions for an optimal such censor to exist. Then, we propose algorithms for computing optimal obstruction censors for knowledge bases with OWL 2 QL as well as restricted OWL 2 RL ontologies, which are based on obstructions of polynomial size.

2 Preliminaries

We adopt standard notions in first-order logic over finite function-free signatures. We treat equality $\approx$ as an ordinary predicate, but assume that any set of formulae Σ contains all the axioms of equality for Σ. A *fact* is a ground, equality-free atom, and a *dataset* is a finite set of facts. A *structure* $\mathcal{I}$ is a pair $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of a domain and interpretation function for the symbols in the signature. We define *homomorphisms* between structures $\mathcal{I}$ and $\mathcal{J}$ in the standard way and $\mathcal{I} \hookrightarrow \mathcal{J}$ denotes the fact that such a homomorphism from $\mathcal{I}$ to $\mathcal{J}$ exists.

(1) $A(x) \wedge R(x,y_1) \wedge B(y_1) \wedge R(x,y_2) \wedge B(y_2) \rightarrow y_1 \approx y_2$,	
(2) $R(x,y) \rightarrow S(x,y)$,	(3) $A(x) \rightarrow \exists y.[R(x,y) \wedge B(y)]$,
(4) $A(x) \rightarrow x \approx a$,	(5) $R(x,y) \wedge S(y,z) \rightarrow T(x,z)$,
(6) $A(x) \rightarrow R(x,x)$,	(7) $A(x) \wedge B(x) \rightarrow C(x)$,
(8) $R(x,x) \rightarrow A(x)$,	(9) $A(x) \wedge R(x,y) \rightarrow B(y)$,
(10) $R(x,y) \rightarrow S(y,x)$,	(11) $R(x,a) \rightarrow B(x)$,
(12) $R(x,y) \rightarrow A(y)$,	(13) $A(x) \rightarrow R(x,a)$,
(14) $A(x) \rightarrow B(x)$,	(15) $R(x,y) \wedge B(y) \rightarrow A(x)$.

Table 1. Horn-$\mathcal{SROIF}$ rules; unary predicates can be $\top$.

A *rule* is a sentence of the form $\forall \boldsymbol{x}.\forall \boldsymbol{z}.[\varphi(\boldsymbol{x},\boldsymbol{z}) \rightarrow \exists \boldsymbol{y}.\psi(\boldsymbol{x},\boldsymbol{y})]$, where $\boldsymbol{x}$, $\boldsymbol{z}$, and $\boldsymbol{y}$ are pairwise disjoint vectors of variables, the *body* $\varphi(\boldsymbol{x},\boldsymbol{z})$ is an equality-free conjunction of atoms with variables $\boldsymbol{x} \cup \boldsymbol{z}$, and the *head* $\psi(\boldsymbol{x},\boldsymbol{y})$ is a conjunction of atoms with variables $\boldsymbol{x} \cup \boldsymbol{y}$. For simplicity universal quantifiers in rules are usually omitted. A rule is *(i) Datalog* if its head consists of a single atom and $\boldsymbol{y}$ is empty; *(ii) guarded* if it has a body atom (a *guard*) mentioning all universally quantified variables; *(iii) linear* if it has a single body atom; and *(iv) multi-linear* if the body contains only guards. An *ontology* is a finite set of rules. We assume that both rule heads and bodies are non-empty and they do not contain the nullary atoms $\top$ and $\bot$. Thus, $\mathcal{O} \cup \mathcal{D}$ is satisfiable for each ontology $\mathcal{O}$ and dataset $\mathcal{D}$, and $\mathcal{O} \not\models \alpha$ for each fact α, which ensures a separation between schema and data.

To capture all OWL 2 profiles, we focus on *Horn-*$\mathcal{SROIF}$. Table 1 provides the normalised axioms of this DL in the form of rules. To capture the semantics of $\top$, usually allowed in DLs, we treat it as a unary predicate and assume that each ontology $\mathcal{O}$ contains the rule $P(x_1,\ldots,x_n) \rightarrow \top(x_i)$ for each predicate P and $1 \leq i \leq n$. A Horn-$\mathcal{SROIF}$ ontology is in *(i) RL* if it has *no* rules of Type (3); *(ii) QL* if it contains *only* rules of Types (2), (3), (10), (12), and (14); *(iii) EL* if it *does not* contain rules of Types (1), (9), and (10).

A *conjunctive query* (*CQ*) is a formula $Q(\boldsymbol{x})$ of the form $\exists \boldsymbol{y}.\varphi(\boldsymbol{x},\boldsymbol{y})$, with $\varphi(\boldsymbol{x},\boldsymbol{y})$ a conjunction of atoms. A *union of CQs* (*UCQ*) is a formula $\bigvee_i Q_i(\boldsymbol{x})$, with each $Q_i(\boldsymbol{x})$ a CQ. A CQ is *Boolean* (*BCQ*) if $\boldsymbol{x}$ is empty. A tuple of constants $\boldsymbol{t}$ is a (*certain*) *answer* to $Q(\boldsymbol{x})$ over ontology $\mathcal{O}$ and dataset $\mathcal{D}$ if $\mathcal{O} \cup \mathcal{D} \models Q(\boldsymbol{t})$. Then, $\mathsf{cert}(Q,\mathcal{O},\mathcal{D})$ is the set of answers to $Q(\boldsymbol{x})$ over $\mathcal{O}$ and $\mathcal{D}$. Given a BCQ Q, $\mathbf{A}[Q]$ denotes the structure interpreting each relation R with $(f(u_1),\ldots f(u_n))$ for every atom $R(u_1,\ldots,u_n)$ in Q, where f maps each constant in Q to itself and each variable y to a fresh element d_y in the structure.

3 Basic Framework

Given an ontology $\mathcal{O}$ and dataset $\mathcal{D}$, we assume that $\mathcal{D}$ is hidden while $\mathcal{O}$ is known to users, who can formulate arbitrary CQs via a query interface. A policy, which

is unknown to users, is given as a set of facts entailed by $\mathcal{O} \cup \mathcal{D}$. It is assumed that system administrators are in charge of specifying policies, and that each policy is assigned to a specific (group of) users by means of standard mechanisms such as role-based access control techniques [17].

Definition 1. *A* policy $\mathcal{P}$ *for* $\mathcal{O}$ *and* $\mathcal{D}$ *is a dataset such that* $\mathcal{O} \cup \mathcal{D} \models \mathcal{P}$. *A* CQE-instance $\mathbf{I}$ *is a triple* $(\mathcal{O}, \mathcal{D}, \mathcal{P})$, *with* $\mathcal{P}$ *a policy for* $\mathcal{O}$ *and* $\mathcal{D}$.

Example 1. Consider the following ontology and dataset that describe an excerpt of a social network including information about movies:

$$\begin{aligned}\mathcal{O}_{\mathsf{ex}} = \{&\mathit{FOf}(x,y) \rightarrow \mathit{FOf}(y,x), \mathit{Susp}(x) \wedge \mathit{Cr}(x) \rightarrow \mathit{Thr}(x),\\ &\mathit{Likes}(x,y) \wedge \mathit{Thr}(y) \rightarrow \mathit{ThrFan}(x)\},\\ \mathcal{D}_{\mathsf{ex}} = \{&\mathit{FOf}(\mathit{John}, \mathit{Bob}),\ \mathit{FOf}(\mathit{Bob}, \mathit{Mary}), \mathit{Likes}(\mathit{John}, \mathit{Seven}),\\ &\mathit{Likes}(\mathit{Bob}, \mathit{Seven}),\ \mathit{Susp}(\mathit{Seven}),\ \mathit{Cr}(\mathit{Seven})\}.\end{aligned}$$

Here, $\mathcal{O}_{\mathsf{ex}}$ states that friendship is symmetric; movies that are both suspense and crime are thrillers; and everyone who likes a thriller is a thriller fan. Assume that John wants to hide his friend list. Then, $\mathcal{P}_{\mathsf{ex}} = \{\alpha_{\mathsf{ex}}\}$ with $\alpha_{\mathsf{ex}} = \mathit{FOf}(\mathit{John}, \mathit{Bob})$, and $\mathbf{I}_{\mathsf{ex}} = (\mathcal{O}_{\mathsf{ex}}, \mathcal{D}_{\mathsf{ex}}, \mathcal{P}_{\mathsf{ex}})$.

A key component of a CQE system is the *censor*, whose goal is to decide according to the policy which query answers can be safely returned to users.

Definition 2. *A* censor *for a CQE-instance* $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ *is a function* cens *that maps each CQ* Q *to a subset of* $\mathsf{cert}(Q, \mathcal{O}, \mathcal{D})$. *The* characteristic theory $\mathsf{Th}_{\mathsf{cens}}$ *of* cens *is the (possibly infinite) set of sentences*

$$\{Q(\boldsymbol{t}) \mid \boldsymbol{t} \in \mathsf{cens}(Q) \text{ and } Q(\boldsymbol{x}) \text{ is a CQ}\}.$$

Censor cens *is* confidentiality preserving *if* $\mathcal{O} \cup \mathsf{Th}_{\mathsf{cens}} \not\models \alpha$ *for each* $\alpha \in \mathcal{P}$. *It is* optimal *if* (i) *it is confidentiality preserving and* (ii) *no confidentiality preserving censor* $\mathsf{cens}' \neq \mathsf{cens}$ *exists such that* $\mathsf{cens}(Q) \subseteq \mathsf{cens}'(Q)$ *for every* Q.

Intuitively, $\mathsf{Th}_{\mathsf{cens}}$ represents the information that a user can potentially gather by asking an unbounded number of queries to the system. If the censor is confidentiality preserving, then no information can be obtained about $\mathcal{P}$, regardless of the CQs asked. Finally, optimal censors maximise information accessibility without compromisig the policy.

4 View and Obstruction Censors

The idea behind view censors is to associate to a CQE instance $\mathbf{I}$ a new dataset, called a *view*. Intuitively, a view encodes the information that a user is allowed to see. The user gets only those query answers that follow from $\mathcal{O}$ and this view. In this way, the main workload of the censor boils down to the computation of certain answers, which can be fully delegated to the query answering engine.[1]

[1] We assume that all the definitions in this section are parameterised by a (fixed) instance $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$.

Definition 3. *The* view censor $\mathsf{vcens}_{\mathbf{I}}^{\mathcal{V}}$ *for* $\mathbf{I}$ *based on a dataset* $\mathcal{V}$ *(a* view*), is the function mapping each CQ* $Q(\boldsymbol{x})$ *to the set* $\mathsf{cert}(Q, \mathcal{O}, \mathcal{D}) \cap \mathsf{cert}(Q, \mathcal{O}, \mathcal{V})$.

Obviously, if we want the censor to enjoy the properties we are after, the view $\mathcal{V}$ must be constructed with care. For the censor to be confidentiality preserving, $\mathcal{O} \cup \mathcal{V}$ must not entail any atom from the policy $\mathcal{P}$, and to be optimal $\mathcal{V}$ must encode as much information from the hidden dataset as possible.

Example 2. Consider a view $\mathcal{V}_{\mathsf{ex}}$ obtained from the dataset $\mathcal{D}_{\mathsf{ex}}$ by replacing *Bob* with a fresh constant an_b. Intuitively, $\mathcal{V}_{\mathsf{ex}}$ is the result of "anonymising" the constant *Bob*, while keeping the structure of the data intact. Since $\mathcal{V}_{\mathsf{ex}}$ contains no information about *Bob*, we have $\mathcal{O}_{\mathsf{ex}} \cup \mathcal{V}_{\mathsf{ex}} \not\models \alpha_{\mathsf{ex}}$, that is the censor based on $\mathcal{V}_{\mathsf{ex}}$ is confidentiality preserving. View $\mathcal{V}_{\mathsf{ex}}$, however, is not optimal: $\mathcal{O}_{\mathsf{ex}} \cup \mathcal{V}_{\mathsf{ex}}$ does not entail the fact $\mathit{Likes}(\mathit{Bob}, \mathit{Seven})$, which is harmless for confidentiality. Indeed, $\mathcal{O}_{\mathsf{ex}} \cup \mathcal{V}'_{\mathsf{ex}} \not\models \alpha_{\mathsf{ex}}$ holds for the extension $\mathcal{V}'_{\mathsf{ex}}$ of $\mathcal{V}_{\mathsf{ex}}$ with $\mathit{Likes}(\mathit{Bob}, \mathit{Seven})$.

View censors require the ability to materialise implicit data, and hence are especially well-suited for RDF-based applications, in which reasoning is performed by a triple store. In OBDA scenarios, however, data is typically managed by an RDBMS and materialisation is not possible. To fulfill the requirement of OBDA applications, we need a different kind of censors.

The idea behind obstruction censors is to associate to $\mathbf{I}$ an *obstruction* in the form of a Boolean UCQ U, such that given a query $Q(\boldsymbol{x})$ and an answer $\boldsymbol{t}$ over $\mathcal{O}$ and $\mathcal{D}$, the censor returns $\boldsymbol{t}$ only if no CQ in U follows from $Q(\boldsymbol{t})$. Thus, the obstruction can be seen as a collection of "forbidden query patterns", which should not be disclosed.

Definition 4. *The* obstruction censor $\mathsf{ocens}_{\mathbf{I}}^{U}$ *for* $\mathbf{I}$ *based on a Boolean UCQ* U *(called an* obstruction*) is the function mapping each CQ* $Q(\boldsymbol{x})$ *to the set*

$$\{\boldsymbol{t} \mid \boldsymbol{t} \in \mathsf{cert}(Q, \mathcal{O}, \mathcal{D}) \text{ and } \mathbf{A}[Q(\boldsymbol{t})] \not\models U\}.$$

Similarly to view censors, obstruction censors do not require dedicated algorithms: checking $\mathbf{A}[Q(\boldsymbol{t})] \models U$ can be delegated to an RDBMS. Also, obstructions can be maintained virtually without the need of data materialisation.

Example 3. The censor based on $\mathcal{V}_{\mathsf{ex}}$ from Example 2 can also be realised with following obstruction U_{ex}:

$$\exists x.\mathit{FOf}(x, \mathit{Bob}) \vee \exists x.\mathit{FOf}(\mathit{Bob}, x) \vee \exists x.\mathit{Likes}(\mathit{Bob}, x) \vee \mathit{ThrFan}(\mathit{Bob}).$$

Intuitively, U_{ex} "blocks" query answers involving *Bob*; and all other answers are the same as over $\mathcal{O}_{\mathsf{ex}} \cup \mathcal{D}_{\mathsf{ex}}$.

As seen in Examples 2 and 3, the same censor may be based on both a view and an obstruction. View and obstruction censors, however, behave rather differently: a view explicitly encodes the information accessible to users, whereas obstructions specify information which users are denied access to. Thus, obstructions are *dual* to views. Unsurprisingly, even in simple cases it is not obvious whether (and how) a view can be realised by an obstruction, or vice-versa.

We next focus on Datalog ontologies and characterise when a given view $\mathcal{V}$ and obstruction U yield the same censor. Each Datalog ontology $\mathcal{O}$ and dataset $\mathcal{D}$ have a unique *least Herbrand model* $\mathcal{H}_{\mathcal{O},\mathcal{D}}$, that is a finite structure that satisfies $\boldsymbol{t} \in \mathsf{cert}(Q, \mathcal{O}, \mathcal{D})$ iff $\mathbf{A}[Q(\boldsymbol{t})] \hookrightarrow \mathcal{H}_{\mathcal{O},\mathcal{D}}$ and hence captures the information relevant to query answering. We can then formalise the duality between views and obstructions in a natural way: U and $\mathcal{V}$ implement the same censor iff U captures the structures *not* homomorphically embeddable into $\mathcal{H}_{\mathcal{O},\mathcal{V}}$. To formalise this statement, we recall the notion of (non-uniform) constraint satisfaction [13].

Definition 5. *Let $\mathcal{J}$ be a finite structure and $\mathcal{C}$ a class of finite structures. The* CSP *of $\mathcal{J}$ relative to $\mathcal{C}$ (denoted $\mathsf{CSP}_{[\mathcal{C}]}(\mathcal{J})$) is the set $\{\mathcal{I} \in \mathcal{C} \mid \mathcal{I} \hookrightarrow \mathcal{J}\}$.*

A central problem is to determine whether a class of finite structures can be captured by a single formula.

Definition 6. *Let $\mathcal{C}$ be a class of finite structures and let $\mathcal{C}' \subseteq \mathcal{C}$. First-order sentence ψ* defines *$\mathcal{C}'$ if $\mathcal{I} \in \mathcal{C}'$ is equivalent to $\mathcal{I} \models \psi$ for every structure $\mathcal{I} \in \mathcal{C}$.*

The correspondence between view and obstruction censors is then as follows.

Theorem 1. *Let $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ be a CQE-instance with $\mathcal{O}$ Datalog ontology, and $\mathcal{C}$ the class of finite structures $\mathcal{I}$ with $\mathcal{I} \hookrightarrow \mathcal{H}_{\mathcal{O},\mathcal{D}}$. Then, $\mathsf{vcens}^{\mathcal{V}}_{\mathbf{I}} = \mathsf{ocens}^{U}_{\mathbf{I}}$ iff U defines $\neg\mathsf{CSP}_{[\mathcal{C}]}(\mathcal{H}_{\mathcal{O},\mathcal{V}})$, for any view $\mathcal{V}$ and obstruction U.*

Using Theorem 1 together with definability results in Finite Model Theory, we can show that views and obstructions cannot simulate one another in general.

Theorem 2. *There is a CQE-instance for which there exists a view censor, but no obstruction censor. There is a CQE-instance for which there exists an obstruction censor, but no view censor.*

5 Optimal View Censors

Our discussion in Section 4 shows that view and obstruction censors should be studied independently. In this section, we focus on view censors.

Before investigating the design of view-based CQE algorithms, we first establish the theoretical limitations of our approach. We show that an optimal view-based censor for an instance $\mathbf{I}$ is not guaranteed to exist since the optimality requirement may lead to infinite "views", even for EL and RL ontologies.

Theorem 3. *There are CQE-instances $\mathbf{I}_1$ and $\mathbf{I}_2$ such that*

- *the ontology of $\mathbf{I}_1$ uses rules of Types* (1) *and* (9)*, and*
- *the ontology of $\mathbf{I}_2$ uses rules of Types* (5)*,* (8) *and* (15)*,*

for which no optimal view censors exist.

The construction of $\mathbf{I}_1$ shows that equality rules lead to non-existence of optimal views; in turn, the construction of $\mathbf{I}_2$ shows that equality is not needed to preclude optimality in the presence of recursion, transitivity axioms, and Self restrictions.

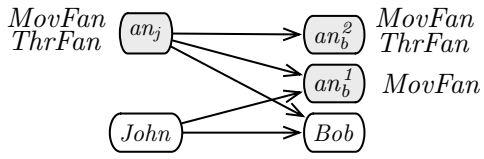

Fig. 1. Essential part of an exhaustive view (solid arrows represent *FOf* relation).

5.1 View Censors for Guarded RL

We next describe how to compute an optimal view for guarded RL ontologies. The idea is to create anonymised copies of constants in the data to encode the information required for optimality. Such a view may use exponentially many anonymised copies of constants.

Definition 7. *Let* $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ *be a CQE-instance with* $\mathcal{O}$ *a guarded RL ontology. A view* $\mathcal{V}$ *consisting of unary atoms* $\mathcal{V}_c^1$ *on constants from* $\mathbf{I}$*, unary atoms* $\mathcal{V}_\exists^1$ *on new constants (anonymised copies), and binary atoms* $\mathcal{V}^2$*, is* exhaustive *on* $\mathbf{I}$ *if it satisfies all of the following.*

1. *The part* $\mathcal{V}_c^1$ *is a maximal set of unary atoms from* $\mathcal{H}_{\mathcal{O},\mathcal{D}}$ *such that* $\mathcal{O} \cup \mathcal{V}_c^1 \not\models \alpha$ *for each* $\alpha \in \mathcal{P}$*.*
2. *For each constant* a *from* $\mathbf{I}$*, and each set* $\mathcal{A}$ *of unary predicates, such that*
   - $A(a) \in \mathcal{H}_{\mathcal{O},\mathcal{D}}$ *for every* $A \in \mathcal{A}$*,*
   - *if* $A, B \in \mathcal{A}$*, and* $\mathcal{O} \models A(x) \wedge B(x) \rightarrow C(x)$*, then* $C \in \mathcal{A}$*,*
   - *if* $A \in \mathcal{A}$*, then* $\mathcal{O} \not\models A(x) \rightarrow x \approx a$ *for each* a*,*
   - $\mathcal{O} \cup \mathcal{V}_c^1 \cup \{A(a') | A \in \mathcal{A}\} \not\models \alpha$ *for each* $\alpha \in \mathcal{P}$ *and a fresh constant* a'*,*

   the view $\mathcal{V}$ *uses a fresh constant* $a_\mathcal{A}$*, and the part* $\mathcal{V}_\exists^1$ *contains all unary atoms* $A(a_\mathcal{A})$ *such that* $A \in \mathcal{A}$*.*
3. *For each constant* a *from* $\mathbf{I}$*, let* σ_a *be the set of constants consisting of* a *itself and all the constants* $a_\mathcal{A}$*. The binary part* $\mathcal{V}^2$ *of the view* $\mathcal{V}$ *contains the atom* $R(a_1^*, a_2^*)$ *for constants* a_1^*, a_2^* *iff*
   - $R(a_1, a_2) \in \mathcal{H}_{\mathcal{O},\mathcal{D}}$*, where* $a_1^* \in \sigma_{a_1}$ *and* $a_2^* \in \sigma_{a_2}$*,*
   - $\mathcal{O} \cup \{R(a_1^*, a_2^*)\} \not\models \alpha$ *for any* $\alpha \in \mathcal{P}$*, and*
   - $\mathcal{O} \cup \mathcal{V}_c^1 \cup \mathcal{V}_\exists^1 \cup \{R(a_1^*, a_2^*)\} \models A(a^*)$ *implies that* $A(a^*) \in \mathcal{V}_c^1 \cup \mathcal{V}_\exists^1$*.*

This definition is constructive and it is routine to devise an algorithm, which for any instance (non-deterministically) constructs an exhaustive view.

Example 4. Consider the following CQE-instance $(\mathcal{O}, \mathcal{D}, \mathcal{P})$:

$\mathcal{O} = \{\mathit{ThrFan}(x) \rightarrow \mathit{MovieFan}(x), \mathit{ThrFan}(y) \wedge \mathit{FOf}(x, y) \rightarrow \mathit{MovieFan}(x)\},$
$\mathcal{D} = \{\mathit{FOf}(\mathit{John}, \mathit{Bob}), \mathit{ThrFan}(\mathit{John}), \mathit{ThrFan}(\mathit{Bob})\},$
$\mathcal{P} = \{\mathit{MovieFan}(\mathit{Bob}), \mathit{MovieFan}(\mathit{John})\}.$

The essential part of the exhaustive view on this CQE-instance is given in Figure 1, where $\mathcal{V}_c^1 = \emptyset$, $\mathcal{V}_\exists^1$ contains unary atoms over the anonymised copies an_b^1, an_b^2 of Bob, and an_j of John, and $\mathcal{V}^2$ contains the depicted binary atoms. Two anonymised copies of Bob are necessary in any optimal view for $\mathbf{I}$ to answer

correctly "harmless" queries like

$$\exists x, y, z.\, \mathit{ThrFan}(z) \land \mathit{MovieFan}(z) \land \mathit{FOf}(y, z) \land \mathit{ThrFan}(y) \land \\ \mathit{MovieFan}(y) \land \mathit{FOf}(y, x) \land \mathit{MovieFan}(x) \land \mathit{FOf}(\mathit{John}, x).$$

The following theorem formulates the desired properties of exhaustive views.

Theorem 4. *Let $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ be a CQE-instance with $\mathcal{O}$ a guarded RL ontology, and $\mathcal{V}$ an exhaustive view on $\mathbf{I}$. Then $\mathcal{V}$ is optimal. Furthermore, if $\mathcal{O}$ is linear, then $\mathsf{vcens}_{\mathbf{I}}^{\mathcal{V}}$ is the only optimal censor for $\mathbf{I}$.*

The proof relies on the following facts. First, the construction ensures that $\mathcal{H}_{\mathcal{O},\mathcal{V}} = \mathcal{V}$ for any exhaustive view $\mathcal{V}$, that is, no rules are applicable to $\mathcal{V}$. Also, properties of $\mathcal{V}_c^1$, $\mathcal{V}_\exists^1$, and $\mathcal{V}^2$ guarantee that $\mathcal{V}$ does not entail any policy atom. Optimality follows from the fact that for any a in $\mathbf{I}$, each combination of its unary atoms that satisfies the relevant axioms in $\mathcal{O}$ is "witnessed" by a new constant from σ_a, and all possible binary atoms which are compatible with those combinations are added to the view. Then, no essentially new atom can be "added" to the view $\mathcal{V}$ without disclosing a policy. The uniqueness of the optimal censor for linear ontologies follows from the lack of choices in the construction of $\mathcal{V}$. An exhaustive view may use exponentially many constants. However, for multi-linear ontologies, optimal views are of polynomial size.

Proposition 1. *Let $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ be a CQE-instance with $\mathcal{O}$ a multi-linear RL ontology. There is an optimal censor for $\mathbf{I}$ based on a view of polynomial size.*

5.2 View Censors for EL and QL

In contrast to OWL 2 RL, the QL and EL profiles can capture existentially quantified knowledge. To bridge this gap, we show that, under some mild conditions, we can transform an ontology $\mathcal{O}$ into a Datalog ontology $\mathcal{O}'$ such that an optimal view for $(\mathcal{O}, \mathcal{D}, \mathcal{P})$ can be directly obtained from such a view for $(\mathcal{O}', \mathcal{D}, \mathcal{P})$. Thus, devising an optimal view censor for an instance is reduced to devising one for an instance with a Datalog ontology.

Definition 8. *Let σ be a set of constants. A Datalog ontology $\mathcal{O}'$ is a (*Datalog*) σ-*rewriting *of an ontology $\mathcal{O}$ if for each fact β and dataset $\mathcal{D}$ over constants from σ we have that $\mathcal{O} \cup \mathcal{D} \models \beta$ iff $\mathcal{O}' \cup \mathcal{D} \models \beta$.*

Proposition 2. *Let $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ be a CQE-instance with $\mathcal{D}$ using set of constants σ, $\mathcal{O}'$ a σ-rewriting of $\mathcal{O}$ such that $\mathcal{O}' \models \mathcal{O}$, and $\mathcal{V}'$ an optimal view for $(\mathcal{O}', \mathcal{D}, \mathcal{P})$. Then $\mathcal{H}_{\mathcal{O}',\mathcal{V}'}$ is an optimal view for $\mathbf{I}$.*

Now, we just need to transform a QL (or guarded EL) ontology into a stronger guarded RL ontology, which, however, entails the same facts for any dataset. We exploit techniques developed for the *combined approach* to query answering [14–16, 19]. The idea is to transform rules of Type (3) into Datalog by Skolemising existentially quantified variables into globally fresh constants. Such transformation strengthens the ontology; however, if applied to a QL or guarded EL ontology, it preserves entailment of facts for any dataset over σ [19].

Definition 9. *Let $\mathcal{O}$ be an ontology and σ a set of constants. The ontology $\Xi_\sigma(\mathcal{O})$ is obtained from $\mathcal{O}$ by replacing each rule of the form $A(x) \to \exists y.[R(x,y) \wedge B(y)]$ with $A(x) \to P(x,a), P(x,y) \to R(x,y), P(x,y) \to B(y)$, where P is a fresh predicate and a is a globally fresh constant not from σ, unique to A and R.*[2]

Proposition 3. *If $\mathcal{O}$ is a Horn-$\mathcal{SROIF}$ ontology, then $\Xi_\sigma(\mathcal{O}) \models \mathcal{O}$. If also $\mathcal{O}$ is either a QL or guarded EL ontology, then $\Xi_\sigma(\mathcal{O})$ is a σ-rewriting of $\mathcal{O}$.*

Propositions 2 and 3 ensure that $\mathcal{H}_{\Xi_\sigma(\mathcal{O}),\mathcal{V}}$ is optimal for $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ with $\mathcal{O}$ a QL or guarded EL ontology, whenever $\mathcal{V}$ is such a view for $(\Xi_\sigma(\mathcal{O}), \mathcal{D}, \mathcal{P})$. The transformation of $\mathcal{O}$ to $\Xi_\sigma(\mathcal{O})$ preserves linearity and guardedness, so $\Xi_\sigma(\mathcal{O})$ is a guarded RL ontology, and the results of Section 5.1 are applicable.

Theorem 5. *A CQE-instance with a QL or guarded EL ontology has an optimal view censor. For QL, it is unique and can be based on a polynomial size view.*

6 Obstruction Censors

We start our study of obstruction censors by focusing on Datalog ontologies and characterising optimality in terms of resolution proofs of the policy. To this end, we first recapitulate the standard notions on (clause) SLD resolution.

Definition 10. *A* goal *is a conjunction of atoms. The* SLD resolution step *takes a goal $\beta_1 \wedge \ldots \wedge \beta_m$ and a Datalog rule $\bigwedge_{i=0}^{k} \gamma_i \to \delta$ and produces a new goal $(\bigwedge_{i=0}^{k} \gamma_i\theta) \wedge \beta_2\theta \wedge \ldots \wedge \beta_m\theta$, where θ is a most general unifier (MGU) of β_1 and δ. A* proof *of a goal G_0 in a Datalog ontology $\mathcal{O}$ and dataset $\mathcal{D}$ is a sequence $G_0 \xrightarrow{r_1,\theta_1} G_1 \xrightarrow{r_2,\theta_2} \ldots \xrightarrow{r_n,\theta_n} G_n$, where $G_n = \top$ and the goal G_i is obtained from the goal G_{i-1} and sentence $r_i \in \mathcal{O} \cup \mathcal{D}$ by an SLD resolution step with MGU θ_i.*

SLD resolution is sound and complete: for each satisfiable $\mathcal{O} \cup \mathcal{D}$ and goal G, a proof of G exists in $\mathcal{O} \cup \mathcal{D}$ iff $\mathcal{O} \cup \mathcal{D} \models \exists^* G$, with $\exists^* G$ the existential closure of G. We next provide a characterisation of optimality based on proofs. Consider a policy atom $\alpha \in \mathcal{P}$ and some proof π of α in $\mathcal{O} \cup \mathcal{D}$. If a censor answers positively sufficiently many BCQs $\exists^* G$ for goals G in π, then a user could "reconstruct" (a part of) π and compromise the policy. Also, there can be many proofs of α, and a user can compromise the policy by reconstructing any of them. Thus, to ensure that a censor is confidentiality preserving, we must guarantee that the obstruction contains enough CQs to prevent reconstruction of any π. If we want the censor to be optimal, the obstruction should not "block" too many queries. As we will see later on, these requirements may be in conflict and lead to an infinite "obstruction". To formalise this intuition we need an auxiliary notion. A *core* of a set of Boolean CQs $\mathbb{Q}$ is a minimal subset $\mathbb{C}$ of $\mathbb{Q}$ such that for each $Q \in \mathbb{Q}$ there exists $Q' \in \mathbb{C}$ with $Q \models Q'$.

[2] To correctly deal with Self restrictions (rules (6) and (8)) a slightly more complex transformation is required. These changes are straightforward but require introducing further notation, so we present here only the basic transformation for simplicity.

Definition 11. *Given a CQE-instance* $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ *with* $\mathcal{O}$ *a Datalog ontology, let* $\mathbb{Q}(\mathbf{I})$ *be the set of all Boolean CQs* $\exists^* G$ *with* $G \neq \top$ *a goal in a proof of a fact* $\alpha \in \mathcal{P}$ *in* $\mathcal{O} \cup \mathcal{D}$*, and let* $\mathbb{S}$ *be a maximal subset of* $\mathbb{Q}(\mathbf{I})$ *such that* $\mathcal{O} \cup \mathbb{S} \not\models \alpha$ *for any* $\alpha \in \mathcal{P}$*. Then, a* pseudo-obstruction Υ *of* $\mathbf{I}$ *is a core of* $\mathbb{Q}(\mathbf{I}) \setminus \mathbb{S}$*.*

We now relate pseudo-obstructions and optimality.

Theorem 6. *Let* $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ *be a CQE-instance with* $\mathcal{O}$ *a Datalog ontology.*

1. *If* Υ *is a finite pseudo-obstruction for* $\mathbf{I}$*, then* $U = \bigvee_{Q \in \Upsilon} Q$ *is an optimal obstruction for* $\mathbf{I}$*.*
2. *If each pseudo-obstruction for* $\mathbf{I}$ *is infinite, then no optimal obstruction censor for* $\mathbf{I}$ *exists.*

This theorem has consequences on the expressive power of obstructions. Using the results from Section 5.1 we can see that optimal view and obstruction censors are incomparable. This complements Theorem 2, which talks about not necessarily optimal censors.

Theorem 7. *There is a CQE-instance with ontology in both RL and EL (respectively, RL) for which an optimal view (respectively, obstruction) censor exists, but no optimal obstruction (respectively, view) censor exists.*

Next, we show how to apply resolution-based techniques to compute optimal obstructions for instances with linear RL ontologies. These results are then adapted to the case of QL. The algorithm for linear RL is based on the computation of the set $\mathbb{Q}(\mathbf{I})$. To do this computation efficient, we need the following auxiliary structure.

Definition 12. *Let* $\mathcal{O}$ *be a linear RL ontology,* $\mathcal{D}$ *a dataset,* x *and* y *fresh variables, and* $\mathcal{A}$ *the set of all equality-free atoms over the signature of* $\mathcal{O} \cup \mathcal{D}$ *extended with* x *and* y*. The* proof graph *of* $\mathcal{O} \cup \mathcal{D}$ *is the directed graph with the set of nodes* $\mathcal{A} \cup \{\top\}$*, and edges* (β, γ) *such that* γ *can be derived from* β *by means of a single SLD resolution step with a rule from* $\mathcal{O} \cup \mathcal{D}$*.*

The following example illustrates proof graphs.

Example 5. Consider a CQE-instance $\mathbf{I}^1_{\mathsf{ex}}$ with ontology $\mathcal{O}^1_{\mathsf{ex}} = \{\mathit{Likes}(x, y) \rightarrow \mathit{Movie}(y),\ \mathit{Likes}(x, y) \rightarrow \mathit{MovieFan}(x)\}$, dataset $\mathcal{D}^1_{\mathsf{ex}} = \{\mathit{Likes}(\mathit{John}, \mathit{Seven})\}$, and the policy of single atom $\alpha^1_{\mathsf{ex}} = \mathit{MovieFan}(\mathit{John})$. A fragment of the proof graph is given in Figure 2.

Using proof graphs we can compute optimal censors.

Theorem 8. *Let* $\mathbf{I} = (\mathcal{O}, \mathcal{D}, \mathcal{P})$ *be a CQE-instance with* $\mathcal{O}$ *a linear RL ontology. For each* $\alpha \in \mathcal{P}$*, let* S_α *be the set of nodes in the proof graph of* $\mathcal{O} \cup \mathcal{D}$ *in a path from* α *to* $\top$*. Finally, let* U *be the Boolean UCQ*

$$\bigvee_{\alpha \in \mathcal{P}} \bigvee_{G \in S_\alpha \setminus \{\top\}} \exists^* G.$$

Then, $\mathsf{ocens}^U_{\mathbf{I}}$ *is the unique optimal censor for* $\mathbf{I}$*, and* U *can be computed in polynomial time in the size of* $\mathbf{I}$*.*

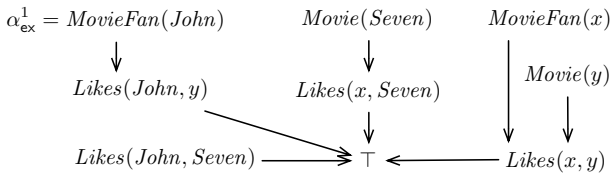

Fig. 2. Fragment of proof graph for $\mathcal{O}^1_{\mathsf{ex}} \cup \mathcal{D}^1_{\mathsf{ex}}$

Example 6. For $\mathbf{I}^1_{\mathsf{ex}}$ from Example 5, there is only one path in the proof graph from α^1_{ex} to $\top$ and $S_{\alpha^1_{\mathsf{ex}}} \setminus \{\top\} = \{MovieFan(John), likes(John, y)\}$. Thus, $U = MovieFan(John) \vee \exists y.Likes(John, y)$ is optimal.

Finally, note that the transformation of a QL ontology $\mathcal{O}$ to an RL ontology $\Xi_\sigma(\mathcal{O})$ given in Definition 9, preserves linearity of rules. Hence, Proposition 3 and Theorem 8 yield the following result.

Theorem 9. *For every CQE-instance with a QL ontology there exists a unique optimal obstruction censor.*

7 Discussion and Conclusions

We have studied CQE in the context of ontologies. Our results yield a flexible way for system designers to ensure selective access to data and provide insights on the fundamental tradeoff between accessibility and confidentiality of information. We have proposed algorithms applicable to the profiles of OWL 2, which can be implemented using off-the-shelf query answering infrastructure. Thus, our algorithms provide a starting point to the development of CQE systems.

The problems studied here remain rather unexplored and we see many open questions. From a theoretic point of view, we plan to consider policies beyond sets of facts (e.g., given as CQs). We also plan to study weaker notions of optimality that can ensure polynomiality of views and obstructions for more expressive languages. From a practical perspective, we will implement our algorithms and test their scalability using state-of-the art Datalog engines such as RDFox.[3]

The approach closest to ours is the view-based access authorisation framework in [9]. In this setting, policies are represented as *authorisation views*: CQs that define the only information accessible to the user; since queries are answered faithfully against the views, there is no explicit notion of policy violation. In contrast, in our setting policies express inaccessible information, and our goal is to maximally answer queries without violating the policy.

Acknowledgements. Work supported by the Royal Society, the EPSRC projects Score!, Exoda, and MaSI^3, and the FP7 project OPTIQUE.

[3] http://www.cs.ox.ac.uk/isg/tools/RDFox/

References

1. Bao, J., Slutzki, G., Honavar, V.: Privacy-Preserving Reasoning on the Semantic Web. In: WI. pp. 791–797. IEEE Computer Society (2007)
2. Biskup, J., Bonatti, P.: Controlled Query Evaluation with Open Queries for a Decidable Relational Submodel. Ann. Math. and Artif. Intell. 50(1-2), 39–77 (2007)
3. Biskup, J., Bonatti, P.A.: Lying Versus Refusal for Known Potential Secrets. Data Knowl. Eng. 38(2), 199–222 (2001)
4. Biskup, J., Bonatti, P.A.: Controlled Query Evaluation for Enforcing Confidentiality in Complete Information Systems. Int. J. Inf. Sec. 3(1), 14–27 (2004)
5. Biskup, J., Weibert, T.: Keeping Secrets in Incomplete Databases. Int. J. Inf. Sec. 7(3), 199–217 (2008)
6. Bonatti, P.A., Kraus, S., Subrahmanian, V.S.: Foundations of Secure Deductive Databases. IEEE Trans. Knowl. Data Eng. 7(3), 406–422 (1995)
7. Bonatti, P.A., Sauro, L.: A Confidentiality Model for Ontologies. In: ISWC. pp. 17–32 (2013)
8. Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: View-based Query Answering over Description Logic Ontologies. In: KR. AAAI Press (2008)
9. Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: View-based Query Answering in Description Logics: Semantics and Complexity. J. Comput. Syst. Sci. 78(1), 26–46 (2012)
10. Cuenca Grau, B.: Privacy in ontology-based information systems: A pending matter. Semantic Web 1(1-2), 137–141 (2010)
11. Cuenca Grau, B., Kharlamov, E., Kostylev, E.V., Zheleznyakov, D.: Controlled Query Evaluation over OWL 2 RL Ontologies. In: ISWC. pp. 49–65 (2013)
12. Cuenca Grau, B., Motik, B.: Reasoning over Ontologies with Hidden Content: The Import-by-Query Approach. J. Artif. Intell. Res. 45, 197–255 (2012)
13. Kolaitis, P.G., Vardi, M.Y.: A Logical Approach to Constraint Satisfaction. In: Complexity of Constraints. pp. 125–155 (2008)
14. Kontchakov, R., Lutz, C., Toman, D., Wolter, F., Zakharyaschev, M.: The Combined Approach to Ontology-Based Data Access. In: IJCAI. pp. 2656–2661 (2011)
15. Lutz, C., Seylan, I., Toman, D., Wolter, F.: The Combined Approach to OBDA: Taming Role Hierarchies Using Filters. In: ISWC. pp. 314–330 (2013)
16. Lutz, C., Toman, D., Wolter, F.: Conjunctive Query Answering in the Description Logic EL Using a Relational Database System. In: IJCAI. pp. 2070–2075 (2009)
17. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-Based Access Control Models. IEEE Computer 29(2), 38–47 (1996)
18. Sicherman, G.L., de Jonge, W., van de Riet, R.P.: Answering Queries Without Revealing Secrets. ACM Trans. Database Syst. 8(1), 41–59 (1983)
19. Stefanoni, G., Motik, B., Horrocks, I.: Introducing Nominals to the Combined Query Answering Approaches for EL. In: AAAI (2013)
20. Stouppa, P., Studer, T.: A Formal Model of Data Privacy. In: PSI (2007)
21. Tao, J., Slutzki, G., Honavar, V.: Secrecy-Preserving Query Answering for Instance Checking in $\mathcal{EL}$. In: RR. pp. 195–203 (2010)