Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)
Main Track

Balancing Expressiveness and Inexpressiveness in View Design

Michael Benedikt[1] , **Pierre Bourhis**[2] , **Louis Jachiet**[3] , **Efthymia Tsamoura**[4]
[1]University of Oxford
[2]CRIStAL, CNRS, University of Lille & INRIA
[3]LTCI, IP Paris
[4]Samsung AI Research
michael.benedikt@cs.ox.ac.uk, pierre.bourhis@inria.fr, louis.jachiet@telecom-paris.fr,
efi.tsamoura@samsung.com

Abstract

We study the design of data publishing mechanisms that allow a collection of autonomous distributed datasources to collaborate to support queries. A common mechanism for data publishing is via *views*: functions that expose derived data to users, usually specified as declarative queries. Our autonomy assumption is that the views must be on individual sources, but with the intention of supporting integrated queries. In deciding what data to expose to users, two considerations must be balanced. The views must be sufficiently expressive to support queries that users want to ask – the *utility* of the publishing mechanism. But there may also be some expressiveness restriction. Here we consider two restrictions, a *minimal information* requirement, saying that the views should reveal as little as possible while supporting the utility query, and a *non-disclosure* requirement, formalizing the need to prevent external users from computing information that data owners do not want revealed. We investigate the problem of designing views that satisfy both an expressiveness and an inexpressiveness requirement, for views in a restricted declarative language (conjunctive queries), and for arbitrary views.

1 Introduction

The value of data is increased when data owners make their data available through publicly-accessible interfaces. The value is magnified even further when multiple data owners publish information from related datasets; this allows users to answer queries that require linking information across datasources.

But the benefits of data publishing come with a corresponding risk of revealing too much. For example, there may be information that a data owner wishes to protect, and a user may be able to infer this information either from the published data in isolation, or from the data published by all parties as a whole. There is thus a need to provide data publishing mechanisms that are simultaneously expressive enough to be useful – they enable users to answer appropriate queries – while satisfying some expressiveness restriction.

In data publishing much of the focus has been on disclosure via the familiar mechanism of *views* – declarative queries whose output is made available to users as a table. In this context the competing requirements on a publishing mechanism have been primarily studied in isolation. There is extensive work on analysis of the utility of a set of views: namely given a query, can it be answered using the views, see, e.g. (Halevy 2001; Calvanese et al. 2012). There has also been research concerning analysis of whether a given set of views obeys some expressiveness *restriction*. The negation of answerability is clearly too weak a restriction, since it just guarantees that on some instance the query answer can not be computed from the view images. A relevant query-based notion of expressiveness restriction is *data-independent privacy*: given the views and a set of "secret" queries, check that the secret query answers can not be computed from the views on any source data. Variants of this problem have been studied in (Nash and Deutsch 2007; Benedikt et al. 2016; Benedikt, Cuenca Grau, and Kostylev 2018; Benedikt et al. 2019). Expressiveness restrictions with a similar flavor have also been studied in the context of ontologies (Bonatti and Sauro 2013). But the question of whether there is an expressiveness restriction for views that does not require the specification of a particular set of secrets, as well as the question of how one obtains views that satisfy both expressiveness and inexpressiveness requirements, has not been considered in the context of traditional queries and views, to the best of our knowledge.

A larger body of work comes from privacy research, considering the design of mechanisms achieving a mix of expressiveness ("utility") and inexpressiveness ("privacy") goals. But the focus is on probabilistic transformations or, more generally, probabilistic protocols (see e.g. (Chaum, Crépeau, and Damgard 1988; Dwork 2006; Dwork and Roth 2014)). The guarantees are probabilistic, sometimes alternatively or additionally with computational restrictions on an external party. Recent efforts (Li et al. 2017) have considered a family of mechanisms that look at database queries, with the utility of a mechanism defined (as in our case) using the notion of query determinacy. But randomness still plays a central role in the mechanism and in the definition of privacy.

In contrast, we consider the question of designing views that use *traditional database queries*, with no randomization, so that conflicting requirements of expressiveness and inexpressiveness are satisfied. Both our expressiveness and inexpressiveness requirements will be given in terms of *exact information-theoretic criteria*: they will be defined in terms of what queries can be answered exactly (as opposed to probabilistically) by a party with unlimited computation

power. Due both to the difference in the mechanisms we consider and the requirements we impose, our contribution has a very different flavor from prior lines of work.

Example 1. *A health study hosted by a government agency holds information about certain treatments, with an abstraction of the data being a database with schema* $\mathsf{Trtmnt}(\mathsf{pid}, \mathsf{tinfo}, \mathsf{tdate})$*, where* pid *might be a national insurance number.*

Demographic information about patients is stored by another agency, in a table $\mathsf{Patient}(\mathsf{pid}, \mathsf{age}, \mathsf{address})$*. The agencies are completely autonomous, perhaps even in distinct administrative regions. But they want to co-operate to support certain queries over the data that legitimate researchers might wish; for example about the relationships between treatment and age:*

$$Q = \exists \mathsf{pid}, \mathsf{address}, \mathsf{tdate}.\mathsf{Trtmnt}(\mathsf{pid}, \mathsf{tinfo}, \mathsf{tdate}) \\ \wedge \mathsf{Patient}(\mathsf{pid}, \mathsf{age}, \mathsf{address})$$

Of course, the parties could agree to an encryption scheme on the patient identifiers, and then expose encrypted versions of their local data. But this would require both strong co-operation of the parties, and the use of views beyond traditional database queries.

But assuming that the parties are restricted to using traditional queries, what is the most restrictive thing that they can do while supporting the ability to answer Q*? Intuitively, the most restrictive views would correspond to one party revealing the projection of the* Trtmnt *table on* pid *and* tinfo *and the other revealing the projection of the* Patient *table on* pid *and* address*.*

If this intuition is correct, it would imply that nothing the parties can do with traditional queries can avoid revealing which patients had particular treatments. That is, they have no choice but to allow an external party to learn the answer to query

$$p = \exists \mathsf{tdate}.\mathsf{Trtmnt}(\mathsf{pid}, \mathsf{tinfo}, \mathsf{tdate})$$

Our results will validate this obvious answer – in this example, the projections described above are the CQ views that reveal minimal information, and one can not support the disclosure of the join query while protecting a query on these join attributes. Further we will show that even using encryption – indeed, even using any deterministic function – the parties can not not reveal less information while supporting exact answering of the query Q*. In contrast, we will also show that in some cases the parties can obtain combinations of expressiveness and inexpressiveness requirements by using counter-intuitive view combinations.*

Our goal here is to look at the problem of designing independent views over multiple relational datasources that satisfy both expressiveness and inexpressiveness requirements. Our expressiveness requirement (usefulness) will be phrased in terms of the ability to answer a relational query, where answering is the traditional deterministic notion used in database theory and knowledge representation. For expressiveness limitations we require the views to be useful but to *minimize information* within a class of views. We also consider an expressiveness limitation specified by *non-disclosure* of a set of *secret queries*. Our contributions include formalizing these notions, characterizing when minimal information views exist and what form they take, and determining when views exist that satisfy utility and expressiveness limitations. We look at these problems both for views given in the standard view language of conjunctive queries, and for arbitrary views. We also consider the impact of background knowledge on these problems.

Organization. Section 2 gives database and logic preliminaries, and then goes on to formalize our expressiveness and inexpressiveness requirements. Section 3 deals with the variant of the problem where the only views considered are conjunctive query views, while Section 4 shows how the situation changes when arbitrary views can be utilized. Section 5 contains extensions when background knowledge is present. Section 5 consider background knowledge about connections across sources. We close with conclusions in Section 6. Full proofs can be found on arXiv.

2 Preliminaries

2.1 Basic Definitions

The bulk of this subsection reviews standard definitions from databases and knowledge representation. But it includes two notions, DCQs and distributed schemas, that are less standard.

Databases and queries. A *schema* consists of a finite set of relations, each with an associated arity (a non-negative number). An *instance* of a schema is an assignment of each relation in the schema of arity n to a collection (finite or infinite) of n-tuples of elements. Given an instance $\mathcal{I}$ and a relation R, we let $\mathcal{I}(R)$ be the n-tuples assigned to R in $\mathcal{I}$. The *active domain* of an instance $\mathcal{I}$, denoted $\mathsf{adom}(\mathcal{I})$, is the set of elements occurring in some tuple of some $\mathcal{I}(R)$. A *fact* consists of a relation R of arity n and an n-tuple $\mathbf{t}$. We write such a fact as $R(\mathbf{t})$. An instance can equivalently be thought of as a set of facts.

An n-ary *query* is a function from instances of a fixed schema $\mathcal{S}$ to some set. We refer to $\mathcal{S}$ as the *input schema* of the query. A *Conjunctive Query* (CQ) is a formula of the form $\exists \mathbf{y} \bigwedge_i A_i$ where A_i are atoms from the schema. A Boolean CQ (BCQ) is a CQ with no free variables. Following the notation used in several places in the literature (e.g. (Arenas, Barceló, and Reutter 2011)) we define a *union of CQs* (UCQ) to be a disjunction of CQs satisfying the *safety condition* where the free variables of each disjunct are the same. Disjunctions of CQs where the safety condition is dropped will play a key role in this paper. The terminology for such queries is less standardized, but we refer to them as *disjunctions of CQs* (DCQs). UCQs can be extended to *relational algebra*, the standard algebraic presentation of first-order relational queries: queries are build up from relation symbols by union, difference, selection, and product (Abiteboul, Hull, and Vianu 1995).

For a logical formula ρ with free variables $x_1, \ldots, x_n$ and an instance $\mathcal{I}$, a *variable binding* σ for ρ in $\mathcal{I}$ is a mapping taking each x_i to an element of $\mathsf{adom}(\mathcal{I})$. We can apply σ to ρ to get a new formula $\sigma(\rho)$ where each x is re-

placed by $\sigma(x)$. Assuming an ordering of the free variables as $x_1, \ldots, x_n$ we may identify a k-tuple $\mathbf{t}$, writing $\rho(\mathbf{t})$ to mean that t_i is substituted for x_i in ρ.

A *homomorphism* between instances $\mathcal{I}_1$ and $\mathcal{I}_2$ is a function f from $\mathsf{adom}(\mathcal{I}_1)$ to $\mathsf{adom}(\mathcal{I}_2)$ such that $R(c_1, \ldots, c_m) \in \mathcal{I}_1$ implies $R(f(c_1), \ldots, f(c_m)) \in \mathcal{I}_2$.

The *canonical database* of a CQ Q, denoted $\mathsf{canondb}(Q)$, is the instance whose facts are the atoms of Q, where each variable v corresponds to an element c_v. The notion of homomorphism from a CQ Q to an instance $\mathcal{I}$ is just a homomorphism from $\mathsf{canondb}(Q)$ to $\mathcal{I}$. A homomorphism from a CQ Q to a CQ Q' is just a homomorphism between their canonical databases, where we additionally require that the mapping be the identity on any free variables or constants. The *output* of a CQ Q on an instance $\mathcal{I}$, denoted $Q(\mathcal{I})$ consists of the restrictions to free variables of Q of the homomorphisms of Q to $\mathcal{I}$. The output of a UCQ is defined similarly. We can choose an ordering of the free variables of Q, and can then say that the output of Q on $\mathcal{I}$ consists of n-tuples. We write $\mathcal{I}, \mathbf{t} \models Q$ for an n-tuple $\mathbf{t}$, if $\mathbf{t} \in Q(\mathcal{I})$. We analogously define $\mathcal{I}, \sigma \models Q$ for a variable binding σ. We sometimes refer to a homomorphism of a BCQ into an instance as a *match*. For a logical formula $\rho(\mathbf{x})$ and a tuple of elements $\mathbf{t}$, $\rho(\mathbf{t})$ denotes the formula where each x_i is substituted with t_i.

A CQ Q_0 is a *subquery* of a CQ Q if the atoms of Q_0 are a subset of the atoms of Q and a variable of Q_0 is free in Q_0 if and only if it is free in Q. A *strict subquery* of Q is a subquery of Q that is not Q itself; Q is *minimal* if there is no homomorphism from Q to a strict subquery of Q.

A *view* over a schema $\mathcal{S}$ consists of an n-ary relation V and a corresponding n-ary query Q_V over relations from $\mathcal{S}$. Given a collection of views $\mathcal{V}$ and instance $\mathcal{I}$, the *view-image* of $\mathcal{I}$, denoted $\mathcal{V}(\mathcal{I})$ is the instance that interprets each $\mathsf{V} \in \mathcal{V}$ by $Q_\mathsf{V}(\mathcal{I})$. We thus talk about *CQ views*, *UCQ views*, etc: views defined by formulas within a class.

Distributed data and views. A *distributed schema* (d-schema) $\mathcal{S}$ consists of a finite set of *sources* Srcs, with each source s associated with a *local schema* $\mathcal{S}_s$. We assume that the relations in distinct local schemas are pairwise disjoint. In Example 1 our distributed schema consisted of two sources, one containing Trtmnt and the other containing $\mathsf{Patient}$. A *distributed instance* (d-instance) is an instance of a distributed schema. For a source s, an s-instance is an instance of the local schema $\mathcal{S}_s$. Given a d-instance $\mathcal{D}$, we denote by $\mathcal{D}_s$ the restriction of $\mathcal{D}$ to relations in s. If d-schema $\mathcal{S}$ is the disjoint union of $\mathcal{S}_1$ and $\mathcal{S}_2$, and we have sources $\mathcal{I}_1$ for $\mathcal{S}_1$ and $\mathcal{I}_2$ for $\mathcal{S}_2$, then we use $(\mathcal{I}_1, \mathcal{I}_2)$ to denote the union of $\mathcal{I}_1$ and $\mathcal{I}_2$, which is an instance of $\mathcal{S}$.

For a given d-schema a *distributed view* (d-view) $\mathcal{V}$ is an assignment to each source s of a finite set $\mathcal{V}_s$ of views over its local schema. Note that here is our "autonomy" assumption on the instances: we are free to design views on each local source, but views can not cross sources. We can similarly talk about CQ-based d-views, relational algebra-based d-views, etc.

Tuple Generating Dependencies. Many semantic relationships between relations can be described using the well-known formalism called *Tuple Generating Dependencies* (TGDs), which we now review. These are logical sentences of the form $\forall \mathbf{x}.\lambda \rightarrow \exists \mathbf{y}.\rho$, where λ and ρ are conjunctions of relational atoms. The notion of a formula ρ *holding* in $\mathcal{I}$ (or $\mathcal{I}$ *satisfying* ρ, written $\mathcal{I} \models \rho$) is the standard one in first-order logic. A *trigger* for τ in $\mathcal{I}$ is a homomorphism h of $\lambda(\mathbf{x})$ into $\mathcal{I}$. Moreover, a trigger h for τ is *active* if no extension of h to a homomorphism of $\rho(\mathbf{x}, \mathbf{y})$ into $\mathcal{I}$ exists. Note that a dependency τ is satisfied in $\mathcal{I}$ if there does not exist an active trigger for τ in $\mathcal{I}$.

Let Σ be a set of TGDs, $\mathcal{I}$ be a finite instance, and Q be a BCQ. We write $\mathcal{I} \wedge \Sigma \models Q$ to mean that every instance containing $\mathcal{I}$ and satisfying Σ also satisfies Q. For two CQs Q and Q', we similarly write $Q \wedge \Sigma \models Q'$ to mean that every instance satisfying Q and Σ satisfies Q'.

2.2 Problem Formalization

We now give the key definitions in the paper, capturing our expressiveness requirements ("usefulness") and expressiveness limitations ("minimally informative" and "non-disclosing"). Our expressiveness requirement is via the concept of determinacy (Nash, Segoufin, and Vianu 2010), formalizing the idea that on any instance there is sufficient information in the views to recapture the query.

Definition 1. *Two d-instances $\mathcal{D}$ and $\mathcal{D}'$ are* indistinguishable *by a d-view $\mathcal{V}$ (or just $\mathcal{V}$-indistinguishable) if $V(\mathcal{D}) = V(\mathcal{D}')$ holds, for each view $V \in \mathcal{V}$.*

Since each view $V \in \mathcal{V}_s$ is defined over relations occurring only in $\mathcal{V}_s$, we can equivalently say that $\mathcal{D}$ and $\mathcal{D}'$ are $\mathcal{V}$-indistinguishable if $V(\mathcal{D}_s) = V(\mathcal{D}'_s)$ holds, for each $V \in \mathcal{V}_s$.

Definition 2. *A d-view $\mathcal{V}$* determines *a query Q* at *a d-instance $\mathcal{D}$ if $Q(\mathcal{D}') = Q(\mathcal{D})$ holds, for each $\mathcal{D}'$ that is $\mathcal{V}$-indistinguishable from $\mathcal{D}$.*

The d-view $\mathcal{V}$ is useful *for Q if $\mathcal{V}$ determines Q on every d-instance (for short, just "$\mathcal{V}$ determines Q").*

Usefulness for a given query Q will be our expressiveness requirement on d-views. Our first inexpressiveness requirement captures the idea that we want to reveal as little as possible:

Definition 3 (Minimally informative useful views)**.** *Given a class of views $\mathcal{C}$ and a query Q, we say that a d-view $\mathcal{V}$ is* minimally informative useful d-view for Q within $\mathcal{C}$ *("Min.Inf. d-view") if $\mathcal{V}$ is useful for Q and for any other d-view $\mathcal{V}'$ useful for Q based on views in $\mathcal{C}$, $\mathcal{V}'$ determines the view definition of each view in $\mathcal{V}$.*

We look at another inexpressiveness requirement that requires an external party to not learn about another query.

Definition 4 (Non-disclosure)**.** *A* non-disclosure function *specifies, for each query p, the set of d-views $\mathcal{V}$ that are said to disclose p. We require such a function F to be* determinacy-compatible*: if $\mathcal{V}_2$ discloses p according to F and $\mathcal{V}_1$ determines each view in $\mathcal{V}_2$, then $\mathcal{V}_1$ also discloses p according to F. If $\mathcal{V}$ does not disclose p, we say that $\mathcal{V}$ is non-disclosing for p (relative to the given non-disclosure function).*

When we can find minimally informative useful d-views, this tells us something about non-disclosure, since it is easy to see that if we are looking to design views that are useful and non-disclosing, it suffices to consider minimally informative views, assuming they exist:

Proposition 1. *Suppose $\mathcal{V}$ is a minimally informative useful d-view for Q within $\mathcal{C}$, and there is a d-view based on views in $\mathcal{C}$ that is useful for Q and non-disclosing for p according to non-disclosure function F. Then $\mathcal{V}$ is useful for Q and non-disclosing for p according to F.*

There are many disclosure functions that are determinacy-compatible. But in our examples, our complexity results, and in Section 5, we will focus on a specific non-disclosure function, whose intuition is that an external party "never infers any answers".

Definition 5. *A d-view $\mathcal{V}$ is* universal non-inference non-disclosing *(UN non-disclosing) for a CQ p if for each instance $\mathcal{I}$ and each tuple $\mathbf{t}$ with $\mathcal{I}, \mathbf{t} \models p$, $\mathcal{V}$ does not determine $p(\mathbf{t})$ at $\mathcal{I}$. Otherwise $\mathcal{V}$ is said to be* UN disclosing *for p.*

This non-disclosure function is clearly determinacy-compatible, so Proposition 1 will apply to it. Thus we will be able to utilize UN non-disclosure as a means of showing that certain d-views are *not* minimally informative.

Variations: background knowledge, and finite instances. All of these definitions can be additionally parameterized by background knowledge Σ, consisting of integrity constraints in some logic. Given d-instance $\mathcal{D}$ satisfying Σ and a d-view $\mathcal{V}$, $\mathcal{V}$ *determines* a query Q over the d-schema at $\mathcal{D}$ relative to Σ if: for every $\mathcal{D}'$ *satisfying* Σ that is $\mathcal{V}$-indistinguishable from $\mathcal{D}$, $Q(\mathcal{D}') = Q(\mathcal{D})$. We say that $\mathcal{V}$ is *useful* for a query Q relative to Σ if it determines Q on every d-instance *satisfying* Σ. We say $\mathcal{V}$ is UN non-disclosing for query p with respect to Σ if $\mathcal{V}$ does not determine p *on any d-instance satisfying* Σ.

By default, when we say "every instance", we mean all instances, finite or infinite. There are variations of this problem requiring the quantification in both non-disclosure and utility to be over finite instances. The advantage of dealing with the unrestricted variant of the problem is that for views and queries in relational algebra the determinacy problem is semi-decidable, and is equivalent to the standard notion of rewritability in relational algebra (Nash, Segoufin, and Vianu 2010). In contrast, the finite version is not semi-decidable even for conjunctive queries and views (Gogacz and Marcinkowski 2016). The use of the unrestricted version will also allow us to make use of the chase construction, which will be convenient in Section 3. However, most of the results in the paper do not depend on this design choice: see Section 6 for further discussion of this.

Main problem. We focus on the problem of determining whether minimally informative useful d-views exist for a given query Q and class $\mathcal{C}$, and characterizing such views when they do exist. When minimally informative useful d-views do not exist, we consider the problem of obtaining a d-view that is useful for Q and which minimizes the set of secrets p that are UN disclosed for p. We refer to Q as the *utility query*, and p as the *secret query*.

Discussion. Our formalization of utility of views is *information-theoretic and exact*: a view is useful if a party with access to the view can compute the exact output of the query (as opposed to the correct output with high probability), with no limit on how difficult the computation may be. The generality of this notion will make our negative results stronger. And it turns out the generality will not limit our positive results, since these will be realized by very simply views.

Our formalization of minimally informative views is likewise natural if one seeks an ordering on sets of views measuring the ability to support exact information-theoretic query answering. Our query-based inexpressiveness notion, non-disclosure, gives a way of seeing the impact of minimally informative useful views on protecting information, and it is also based on information-theoretic and exact criteria. We exemplify our general definition of non-disclosure function with UN non-disclosure, which has been studied in prior work under several different names (Nash and Deutsch 2007; Benedikt et al. 2016; Benedikt, Cuenca Grau, and Kostylev 2018; Benedikt et al. 2019). We choose the name "non-disclosing" rather than "private" for all our query-based expressiveness restrictions, since they are clearly very different from more traditional probabilistic privacy guarantees (Dwork and Roth 2014). On the one hand the UN non-disclosure guarantee is weak in that p is considered safe for $\mathcal{V}$ (UN non-disclosed) if an attacker can never infer that p is true with absolutely certainty. Given a distribution on source instances, the information in the views may still increase the likelihood that p holds. On the other hand, UN non-disclosure is quite strong in that it must hold on *every* source instance. Thus, although we do not claim that this captures all intuitively desirable properties of privacy, we do feel that it allows us to explore the ability to create views that simultaneously support the strong ability to answer certain queries in data integration and the strong inability to answer other queries.

Restrictions and simplifications. Although the utility and non-disclosure definitions make sense for any queries, *in this paper we will assume that Q and p are BCQs without constants (abusing notation by dropping "without constants").* While we restrict to the case of a single utility query and secret query here, *all of our results have easy analogs for a finite set of such queries*.

2.3 Some Tools

Throughout the paper we rely on two basic tools.

Canonical views. Recall that we are looking for views that are useful for answering a CQ Q over a d-schema. The "obvious" set of views to try are those obtained by partitioning the atoms of Q among sources, with the free variables of the views including the free variables of Q and the variables occurring in atoms from different sources.

Given a CQ Q over a d-schema, and a source s, we denote by $\mathsf{SVars}(s, Q)$ the variables of Q that appear in an atom from source s. We also denote by $\mathsf{SJVars}(s, Q)$, the "source-join variables of s" in Q: the variables that are in $\mathsf{SVars}(s, Q)$ and which also occur in an atom of another source.

Definition 6. *The* canonical view of Q *for source* s, $\mathsf{CanView}^s(Q)$, *has a view definition formed by conjoining all* s*-source atoms in* Q *and then existentially quantifying all bound variables of* Q *in* $\mathsf{SVars}(s, Q) \setminus \mathsf{SJVars}(s, Q)$. *The* canonical d-view of Q *is formed by taking the canonical view for each source.*

In Example 1, the canonical d-view is what we referred to as "the obvious view design". It would mean that one source has a view exposing $\exists \mathsf{tdate}\ \mathsf{Trtmnt}(\mathsf{pid}, \mathsf{tinfo}, \mathsf{tdate})$ – since pid is a source-join variable while tinfo is a free variable of Q. The other source should expose a view revealing $\exists \mathsf{address}.\mathsf{Patient}(\mathsf{pid}, \mathsf{age}, \mathsf{address})$, since address is neither a free variable nor shared across sources.

The critical instance. In the definition of UN non-disclosure of a query by a set of views, we required that on *any* instance of the sources, a user who has access to the views cannot reconstruct the answer to the query p. An instance that will be helpful in several examples is the following "most problematic" instance (Marnette 2009).

Definition 7. *The* critical instance *of a schema* $\mathcal{S}$ *is the instance whose active domain consists of a single element* $*$ *and whose facts are* $R(*, \ldots, *)$ *for all relational names* R *in* $\mathcal{S}$.

Note that every BCQ over the relevant relations holds on the critical instance of the source. The critical instance is the hardest instance for UN non-disclosure in the following sense:

Theorem 1. *(Benedikt et al. 2016; Benedikt, Cuenca Grau, and Kostylev 2018) Consider any CQ views* $\mathcal{V}$*, and any BCQ* p*. If* p *is determined by* $\mathcal{V}$ *at some instance of the source schema then it is determined by* $\mathcal{V}$ *at the critical instance.*

3 CQ Views

Returning to Example 1, recall the intuition that the canonical d-view of Q is the "least informative d-view" that supports the ability to answer Q. We start our analysis by proving such a result, but with two restrictions: Q must be a minimal CQ, and we only consider views specified by CQs:

Theorem 2. *[Minimally informative useful CQ views] For every minimal BCQ* Q*, the canonical d-view of* Q *is minimally informative. That is, if any CQ-based d-view* $\mathcal{V}$ *determines a* Q*, then* $\mathcal{V}$ *determines each canonical view* $\mathsf{CanView}^s(Q)$ *of* Q.

We only sketch the proof here. The first step is to show that the determinacy of a CQ Q by a CQ-based d-view $\mathcal{V}$ leads to a certain homomorphism of Q to itself. This step follows using a characterization of determinacy of CQ queries by CQ views via the well-known chase procedure (See (Benedikt, ten Cate, and Tsamoura 2016) or the "Green-Red chase" in (Gogacz and Marcinkowski 2015)). The second step is to argue that if this homomorphism is a bijection, then it implies that the canonical d-view determines Q, while if the homomorphism is not bijective, we get a contradiction of minimality. This step relies on an analysis of how the chase characterization of determinacy factorizes over a d-schema.

Consequences. Combining Theorem 2 and Proposition 1 gives a partial answer to the question of how to obtain useful and non-disclosing views:

Corollary 1. *For any non-disclosure function* F*, BCQs* Q *and* p*, if there is a CQ-based d-view that is useful for* Q *and non-disclosing for* p *according to* F*, then the canonical d-view of* Q^{min} *is such a d-view, where* Q^{min} *is any minimal CQ equivalent to* Q.

If we consider the specific non-disclosure notion, UN non-disclosure, we can infer a complexity bound from combining these results with prior work on the complexity of checking non-disclosure (Theorem 44 of (Benedikt, Cuenca Grau, and Kostylev 2018)):

Corollary 2. *There is a* Σ_2^p *algorithm taking as input BCQs* Q *and* p *and determining whether there is a CQ-based d-view that is useful for* Q *and UN non-disclosing for* p*. If* Q *is assumed minimal then the algorithm can be taken to be in* CoNP.

The following example shows that the requirement that Q is minimal (that is, has no redundant conjuncts) in Theorem 2 is essential.

Example 2. *Consider two sources. The first source comprises the relations* R_1*,* R_2 *and* R_3*, while the second source comprises a single relation* T*. Consider also the conjunctions of atoms* C_1 *and* C_2 *defined as:*

$$C_1 = R_1(x, y) \wedge T(x)$$
$$C_2 = R_1(x', y') \wedge R_1(y', z') \wedge R_1(z', x') \wedge T(x') \wedge R_2(y') \wedge R_3(z')$$

The conjunction C_1 *states that there is an element in* T *that is the source of an* R *edge. The conjunction* C_2 *states that there is an* R_1*-triangle with one vertex in* T*, a second in* R_2*, and a third in* R_3*. Consider now the query* Q *defined as*

$$\exists x, y, x', y', z'.\ C_1 \wedge C_2$$

Note that the conjunction of atoms C_1 *in* Q *is redundant. Indeed, the query* $Q^{min} = \exists x', y', z'.C_2$ *is equivalent to* Q.

The canonical view of Q *for the* R_i*'s source is:*

$$\exists y, y', z'.R_1(x, y) \wedge R_1(x', y') \wedge R_1(y', z') \wedge R_1(z', x') \wedge R_2(y') \wedge R_3(z')$$

But the canonical view of Q^{min} *for this source is*

$$\exists y', z'.\ R_1(x', y') \wedge R_1(y', z') \wedge R_1(z', x') \wedge R_2(y') \wedge R_3(z')$$

Theorem 2 tells us that the canonical d-view of Q^{min} *is a minimally informative useful d-view within the class of CQ views. We claim that the canonical d-view of* Q *is* not *minimally informative for this class, and in fact reveals significantly more than the canonical d-view of* Q^{min}*. Consider the secret query* $p = \exists t.R_1(t, t)$ *stating that there is an* R_1 *self-loop. The canonical d-view of* Q^{min} *is UN non-disclosing for* p*. Indeed, given any instance* $\mathcal{D}$*, consider the instance* $\mathcal{D}'$ *in which the* T *source is identical to the one in* $\mathcal{D}$*, but the* R *source is replaced by one where each node* e *in the canonical*

view for $\mathcal{D}$ is in a triangle with distinct elements for y', z'. Such a $\mathcal{D}'$ does not satisfy p, and is indistinguishable from $\mathcal{D}$ according to the canonical d-view of Q^{min}.

In contrast, the canonical d-view of Q is UN disclosing for p. Consider $\mathcal{D}_0$, the critical instance for the source schema (see Section 2). On $\mathcal{D}_0$ the returned bindings have x as only $*$, and from this we can infer that the witness elements for y' and z' can only be $*$, and hence the d-view discloses p with $\mathcal{D}_0$ as the witness.

By Proposition 1 the canonical d-view of Q can not be minimally informative within the class of CQ views.

4 Arbitrary Views

In the previous section we showed that the canonical d-view is minimally informative within the class of CQ views, assuming that the utility query is minimized. We now turn to minimally informative useful views, not restricting to views given by CQ view definitions.

Our goal will be to arrive at a generalization of the notion of canonical d-view that gives the minimal information over arbitrary d-views that are useful for a given BCQ Q. That is we want to arrive at an analog of Theorem 2 replacing "CQ views" by "arbitrary views" and "canonical view" by a generalization.

An equivalence class representation of minimally informative views. Recall that general views are defined by queries, where a query can be any function on instances. An *Equivalence Class Representation of a d-view* (ECR) consists of an equivalence relation $\equiv_s$ for each source s. An ECR is just another way of looking at a d-view defined by a set of arbitrary functions on instances: given a function F, one can define an equivalence relation by identifying two local instances when the values of F are the same. Conversely, given an equivalence relation then one can define a function mapping each instance to its equivalence class. A d-instance $\mathcal{D}$ is indistinguishable from a d-instance $\mathcal{D}'$ by the d-view specified by ECR $\langle \equiv_s\colon s \in \mathsf{Srcs}\rangle$ exactly when $\mathcal{D}_s \equiv_s \mathcal{D}'_s$ holds for each s. Determinacy of one d-view by another corresponds exactly to the refinement relation between the corresponding ECRs. We will thus abuse notation by talking about indistinguishability, usefulness, and minimal informativeness of an ECR, referring to the corresponding d-view.

Our first step will be to show that there is an easy-to-define ECR whose corresponding d-view is minimally informative. For a source s, an *s-context* is an instance for each source other than s. Given an s-context C and an s-instance $\mathcal{I}$, we use $(\mathcal{I}, C)$ to denote the d-instance formed by interpreting the s-relations as in $\mathcal{I}$ and the others as in C.

We say two s-instances $\mathcal{I}, \mathcal{I}'$ are *(s, Q)-equivalent* if for any s-context C, $(\mathcal{I}, C) \models Q \Leftrightarrow (\mathcal{I}', C) \models Q$. We say two d-instances $\mathcal{D}$ and $\mathcal{D}'$ are *globally Q-equivalent* if for each source s, the restrictions of $\mathcal{D}$ and $\mathcal{D}'$ over source s, are (s, Q)-equivalent.

Global Q-equivalence is clearly an ECR. Via "swapping one component at a time" we can see that the corresponding d-view is useful for Q. It is also not difficult to see that this d-view is minimally informative for Q within the class of all views:

Proposition 2. *The d-view corresponding to global Q-equivalence is a minimally informative useful d-view for Q within the collection of all views.*

Note that the result can be seen as an analog of Theorem 2. From it we conclude an analog of Corollary 1:

Proposition 3. *If there is any d-view that is useful for BCQ Q and non-disclosing for BCQ p, then the d-view given by global Q-equivalence is useful for Q and non-disclosing for p.*

From an ECR to a concrete d-view. We now have a useful d-view that is minimally informative within the set of all d-views, but it is given only as the ECR global Q-equivalence, and it is not clear that there are any views in the usual sense – isomorphism-invariant functions mapping into relations of some fixed schema — that correspond to this ECR. Our next goal is to show that global Q-equivalence is induced by a d-view defined using standard database queries.

A *shuffle* of a CQ is a mapping from its free variables to themselves (not necessarily injective). Given a CQ Q and a shuffle μ, we denote by $\mu(Q)$ the CQ that results after replacing each variable occurring in Q by its μ-image. We call $\mu(Q)$ a *shuffled query*. For example, consider the query $\exists y.R(x_1, x_2, x_2, y) \wedge S(x_2, x_3, x_3, y)$. Then, the query $\exists y.R(x_2, x_1, x_1, y) \wedge S(x_1, x_1, x_1, y)$ is a shuffle of Q.

The *canonical context query for Q at source s*, $\mathsf{CanCtxt}^s(Q)$, is the CQ whose atoms are all the atoms of Q that are *not* in source s, and whose free variables are $\mathsf{SJVars}(s, Q)$.

Definition 8. *For a source s, a BCQ Q and a variable binding σ for Q, a shuffle μ of $\mathsf{CanView}^s(Q)$ is* invariant relative to $\langle \sigma, \mathsf{CanCtxt}^s(Q)\rangle$ *if for any d-instance $\mathcal{I}'$ where $\mathcal{I}', \sigma \models \mathsf{CanCtxt}^s(Q)$, we have $\mathcal{I}', \sigma \models \mu(\mathsf{CanCtxt}^s(Q))$.*

Note that we can verify this invariance by finding a homomorphism from $\sigma(\mu(\mathsf{CanCtxt}^s(Q)))$ to $\sigma(\mathsf{CanCtxt}^s(Q))$.

Invariance talks about every binding σ. We would like to abstract to bindings satisfying a set of equalities. A *type* for $\mathsf{SJVars}(s, Q)$ is a set of equalities between variables in $\mathsf{SJVars}(s, Q)$. The notion of a variable binding satisfying a type is the standard one. For a type τ, we can talk about a mapping μ being *invariant relative to* $\langle \tau, \mathsf{CanCtxt}^s(Q)\rangle$: the invariance condition holds for all bindings σ satisfying τ.

For a source s and a CQ Q, let $\tau_1, \ldots, \tau_n$ be all the equality types over the variables in $\mathsf{SJVars}(s, Q)$ and $\mathbf{x}$ be the variables in $\mathsf{SJVars}(s, Q)$.

Definition 9. *The* invariant shuffle views *of Q for source s is the set of views $V_{\tau_1}, \ldots, V_{\tau_n}$ where each V_{τ_i} is defined as $\tau_i(\mathbf{x}) \wedge \bigvee_\mu \mu(\mathsf{CanView}^s(Q))$, where $\mathbf{x}$ are the source-join variables of Q for source s, and where the disjunction is over shuffles invariant relative to τ_i.*

Note that since the domain of μ is finite, there are only finitely many mappings on them, and thus there are finitely many disjuncts in each view up to equivalence.

We can show that global Q equivalence corresponds to agreement on these views:

Proposition 4. *For any BCQ Q and any source s, two s-instances are (s, Q)-equivalent if and only if they agree on each invariant shuffle view of Q for s.*

Putting together Proposition 3 and 4, we obtain:

Theorem 3. *[Minimally informative useful d-views among the class of all views] The invariant shuffle views are minimally informative for Q within the class of all views.*

This yields a corollary for non-disclosure analogous to Corollary 1:

Corollary 3. *If an arbitrary d-view $\mathcal{V}$ is useful for BCQ Q and non-disclosing for BCQ p, then the d-view containing, for each source s, the invariant shuffle views of Q for s, is useful and non-disclosing. In particular, some DCQ is useful for Q and non-disclosing for p.*

In Example 1 there are no nontrivial shuffles, so we can conclude that the canonical d-view is minimally informative within the class of all views. In general the invariant shuffle views can be *unsafe*: different disjuncts may contain distinct variables. Of course, they can be implemented easily by using a wildcard to represent elements outside the active domain. Further, we can convert each of these unsafe views to an "information-equivalent" set of relational algebra views:

Proposition 5. *For every view defined by a DCQ (possibly unsafe), there is a finite set of relational algebra-based views $\mathcal{V}'$ that induces the same ECR. Applying this to the invariant shuffle views for a CQ Q, we can find a relational algebra-based d-view that is minimally informative for Q within the class of all views.*

The intuition behind the proposition is to construct separate views for different subsets of the variables that occur as a CQ disjunct. A view with a given set of variables S will assert that some CQ disjunct with variables S holds and that no disjunct corresponding to a subset of S holds.

Example 3. *Consider a d-schema with two sources, one containing a ternary relation R and the other containing a unary relation S. Consider the utility query Q:*

$$\exists x_1, x_2, y. R(x_1, x_2, y) \wedge R(y, x_2, x_1) \wedge S(x_1) \wedge S(x_2)$$

The canonical view for the R source $\mathsf{CanView}^R(Q)$ is $\exists y.\ R(x_1, x_2, y) \wedge R(y, x_2, x_1)$.

Observe that Q is a minimal CQ, and hence by Theorem 2 the canonical d-view of Q is minimally informative among the CQ-based d-views. We will argue that this d-view is not minimally informative useful for Q among all d-views, by arguing that it discloses more secrets than the shuffle views disclose.

Consider the secret query $p = \exists x. R(x, x, x)$. We show that the canonical d-view of Q is UN disclosing for p. Consider the critical instance of the R-source. An external party will know that the instance contains $\{R(*, *, y_0), R(y_0, *, *)\}$ for some y_0. On the other hand, if $y_0 \neq *$, then the canonical d-view would reveal $x_1 = y_0, x_2 = *$. So y_0 must be $*$, and therefore p is disclosed. By Corollary 1, we know that no CQ-based d-view can be UN non-disclosing for p and useful for Q.

The shuffle views of Q are always useful for Q. We will show that they are UN non-disclosing for p. Let us start by deriving the invariant shuffle views for the R source. There are two types, τ_1 in which $x_1 = x_2$, and τ_2 in which the variables are not identified.

For a binding satisfying τ_1, the canonical view of Q for the R source is equivalent to

$$\exists y. R(x_1, x_1, y) \wedge R(y, x_1, x_1)$$

Since there is only one free variable in it, there is only one invariant shuffle, the identity. Thus

$$\mathsf{V}_{\tau_1} = \exists y. R(x_1, x_1, y) \wedge R(y, x_1, x_1)$$

For bindings satisfying τ_2 there are several shuffles invariant for $\mathsf{CanCtxt}^R(Q) = S(x_1) \wedge S(x_2)$: the identity, the shuffle which swaps x_1 and x_2, the shuffle in which x_1 and x_2 both go to x_1, and the shuffle in which both x_1 and x_2 go to x_2. Thus we get the view V_{τ_2} defined as $x_1 \neq x_2$ conjoined with:

$$\begin{aligned}
&\exists y. R(x_1, x_2, y) \wedge R(y, x_2, x_1) \vee \\
&\exists y. R(x_1, x_1, y) \wedge R(y, x_1, x_1) \vee \\
&\exists y. R(x_2, x_2, y) \wedge R(y, x_2, x_2) \vee \\
&\exists y. R(x_2, x_1, y) \wedge R(y, x_1, x_2)
\end{aligned}$$

This last view is unsafe, but via Proposition 5 we can convert it into a safe relational algebra view that yields the same ECR, $\mathsf{V}^{\mathsf{safe}}_{\tau_2}$ defined as $x_1 \neq x_2$ conjoined with:

$$\begin{aligned}
&\neg(\exists y. R(x_1, x_1, y) \wedge R(y, x_1, x_1)) \wedge \\
&\neg(\exists y. R(x_2, x_2, y) \wedge R(y, x_2, x_2)) \wedge \\
&[(\exists y. R(x_1, x_2, y) \wedge R(y, x_2, x_1) \vee \\
&\exists y. R(x_2, x_1, y) \wedge R(y, x_1, x_2))]
\end{aligned}$$

We now argue that the shuffle views are UN non-disclosing for p. This is because in any d-instance we can replace each fact $R(x_0, x_0, x_0)$ by facts $R(x_0, x_0, c)$ and $R(c, x_0, x_0)$ for a fresh c, obtaining an indistinguishable instance where p does not hold. Hence by Proposition 1, the canonical views of Q can not be minimally informative within the class of all views, or even within the class of relational algebra views.

5 Impact of Background Knowledge

We now look at how the problem of designing views that balance expressiveness and inexpressiveness restrictions changes in the presence of background knowledge on the sources.

Local background knowledge. We start with the case of a background theory Σ in which each sentence is *local*, referencing relations on a single source.

It is easy to show that we can not generalize the prior results for CQ views to arbitrary local background knowledge. Intuitively using such knowledge we can encode design problems for arbitrary views using CQ views.

Thus we restrict to *local constraints Σ that are existential rules*. We show that the results on CQ views extend to this setting. We must now consider utility queries Q that are *minimal with respect to* Σ, meaning that there is no strict subquery equivalent to Q under Σ. By modifying the chase-based approach used to prove Theorem 2, we show:

Theorem 4. *[Min.Inf. CQ views w.r.t. local rules] Let Σ be a set of local existential rules, Q a CQ minimal with respect to Σ. Then the canonical d-view of Q is minimally informative useful within the class of CQ views relative to Σ.*

The analog of Corollary 1 follows from the theorem:

Corollary 4. *If any CQ based d-view is useful for Σ-minimal Q and non-disclosing for p relative to Σ, then the canonical d-view of Q is useful for Q and non-disclosing for p relative to Σ.*

Consequences for decidability. Theorem 4 shows that even in the presence of arbitrary local existential rules Σ it suffices to minimize the utility query under Σ and check the canonical d-view for non-disclosure under Σ. For arbitrary existential rules, even CQ minimization is undecidable. But for well-behaved classes of rules (e.g. those with terminating chase (Cuenca Grau et al. 2013; Baget et al. 2014), or frontier-guarded rules (Baget et al. 2011)) we can perform both minimization and UN non-disclosure checking effectively (Benedikt et al. 2019).

We can also extend the results on arbitrary d-views to account for local existential rules Σ. The notion of a shuffle being Σ-invariant is defined in the obvious way, restricting to instances that satisfy the constraints in Σ. The Σ-invariant shuffle views are also defined analogously; they are DCQ views, but can be replaced by the appropriate relational algebra views. Following the prior template, we can show:

Theorem 5. *[Min.Inf. views w.r.t. local rules] For any set of local existential rules Σ, the Σ-invariant shuffle views of Q provide a minimally informative useful d-view for Q within the class of all views, relative to Σ.*

The result has effective consequences for "tame" rules (e.g. with terminating chase) with no non-trivial invariant shuffles. In such cases, the Σ-invariant shuffle views degenerate to the canonical d-view, and we can check whether the canonical d-view is non-disclosing effectively (Benedikt et al. 2016; Benedikt et al. 2019).

Non-local background knowledge. The simplest kind of non-local constraint is the replication of a table between sources. Unlike local constraints, these require some communication among the sources to enforce. Thus we can consider a replication constraint to be a restricted form of source-to-source communication.

We will see that several new phenomena arise in the presence of replication constraints. Recall that with only local constraints, we have useful d-views with minimal information. We can not guarantee the existence of such a d-view in the presence of replication:

Proposition 6. *There is a schema with replication constraint Σ and a BCQ Q where there is no minimally informative useful d-view for Q within the class of all views w.r.t. Σ.*

Our proof of Proposition 6 uses a schema with unary relations R, S, T. There are two sources: R and T are in different sources, and S is replicated between the two sources. Let Q be $\exists x.R(x) \wedge S(x) \wedge T(x)$.

We will explain how our views act when the active domain of our instances is over the integers. The proof will easily be seen to extend to arbitrary instances (e.g. by having the views reveal all information outside of the integers).

Consider the functions $F_1(x) = 2x + 3$ for x even and $2x + 2$ for x odd. $F_2(x) = 2x + 4$ for x even and $3x$ for x odd. Let Str_1 be the function that applies F_1 to a relation element-wise, and similarly define Str_2 using F_2. Notice that F_1 maps 0 to 3 and 1 to 4, while F_2 maps 0 to 4 and 1 to 3.

We define a d-view $\mathcal{V}_1$ via an ECR, relating two instances $\mathcal{I}$ and $\mathcal{I}'$ of the source exactly when $\mathcal{I}'$ can be obtained from applying Str_1 on $\mathcal{I}$ some number of times (applying it to both relations of the source) or vice versa. That is, the ECR of $\mathcal{V}_1$ is the smallest equivalence relation containing each pair $(\mathcal{I}, \mathsf{Str}_1(\mathcal{I}))$. Let $\mathcal{V}_2$ be defined analogously using Str_2. To see that $\mathcal{V}_1$ and $\mathcal{V}_2$ are useful we will use the following claim, which captures their key properties:

Claim 1. *If we have two d-instances satisfying the replication constraint, $(\mathcal{I}_1, \mathcal{I}_2)$ and $(\mathcal{I}'_1, \mathcal{I}'_2)$, with the replicated relation instances non-empty, then:*

- *We cannot have $\mathcal{I}'_1 = \mathsf{Str}_1^i(\mathcal{I}_1)$, $\mathcal{I}'_2 = \mathsf{Str}_1^j(\mathcal{I}_2)$ with $i \neq j$; and similarly for Str_2.*
- *We cannot have $\mathcal{I}'_1 = \mathsf{Str}_1^i(\mathcal{I}_1)$ and $\mathcal{I}_2 = \mathsf{Str}_1^j(\mathcal{I}'_2)$ unless $i = j = 0$; and similarly for Str_2.*

Proof. Let S be the content of the replicated relation in $(\mathcal{I}_1, \mathcal{I}_2)$, while S' is the content of the replicated relation in $(\mathcal{I}'_1, \mathcal{I}'_2)$.

We focus first on Str_1. For the first item, let $c(S) = max\{|x + 2| \mid x \in S\}$. We can check directly that for any non-empty S, $c(\mathsf{Str}_1(S)) > c(S)$. Then $\mathsf{Str}_1^i(S) = S' = \mathsf{Str}_1^j(S)$ implies that $i = j$ as otherwise $c(\mathsf{Str}_1^i(S)) > c(\mathsf{Str}_1^j(S))$ when $i > j$ and $c(\mathsf{Str}_1^i(S)) < c(\mathsf{Str}_1^j(S))$ when $i < j$. For the second item we would have, $\mathsf{Str}_1^i(S) = S'$ and $\mathsf{Str}_1^j(S') = S$ which means $\mathsf{Str}_1^{i+j}(S) = S$ which is only possible for $i + j = 0$.

For Str_2, the proof is the same, but now using the function d defined as $d(S) = max\{|x+1.5| \mid x \in S\}$. We can check that $d(S) < d(\mathsf{Str}_2(S))$ for any non-empty S.

□

From the claim, usefulness follows easily. Suppose we have $(\mathcal{I}_1, \mathcal{I}_2)$ satisfying Q, and $(\mathcal{I}'_1, \mathcal{I}'_2)$ is equivalent to $(\mathcal{I}_1, \mathcal{I}_2)$. From $(\mathcal{I}_1, \mathcal{I}_2)$ satisfies Q, we know that the replicated relation in $\mathcal{I}_1$ and $\mathcal{I}_2$ is non-empty, so the claim applies to tell us that $\mathcal{I}'_1 = \mathcal{I}_1, \mathcal{I}'_2 = \mathcal{I}_2$.

Now suppose $\mathcal{V}$ were a minimally informative useful d-view for Q. We must have $\mathcal{V}_1$ and $\mathcal{V}_2$ determine $\mathcal{V}$. Thus in particular if we have two local instances that agree on *either* $\mathcal{V}_1$ or $\mathcal{V}_2$, then they agree on $\mathcal{V}$.

Consider a d-instance $\mathcal{D}$ with $R = \{0\}$, $S = \{0, 1\}$, $T = \{0\}$, and a d-instance $\mathcal{D}'$ with $R = \{3\}$, $S = \{3, 4\}$, $T = \{4\}$. These are both valid instances (i.e. the replication constraint is respected). But $\mathcal{D}'$ is obtained from $\mathcal{D}$ by applying Str_1 on the source with R, and by applying Str_2 on the other source.

Thus $\mathcal{D}'$ and $\mathcal{D}$ are indistinguishable by $\mathcal{V}$, but Q has a match in $\mathcal{D}$ but not in $\mathcal{D}'$. This contradicts the assumption that $\mathcal{V}$ is useful for Q.

Given Proposition 6, for the remainder of we will focus on obtaining useful views that minimize the set of queries that are UN non-disclosed. If Q is a CQ, we say that a d-view $\mathcal{V}$ is *UN non-disclosure minimal for* Q within a class of views $\mathcal{C}$ if: $\mathcal{V}$ is useful for Q and for any BCQ p, if there is a d-view based on $\mathcal{C}$ which is useful for Q and UN non-disclosing for p, then $\mathcal{V}$ is UN non-disclosing for p Proposition 1 implies that if $\mathcal{V}$ is Min.Inf. within $\mathcal{C}$, then it is UN non-disclosure minimal for any BCQ Q within $\mathcal{C}$.

We can use the technique in the proof of Proposition 6 to show a more promising new phenomenon: there may be d-views that are useful for CQ Q and UN non-disclosing for CQ p, but they are much more intricate than any query related to the canonical d-view of Q. In fact we can show that with a fully-replicated relation in the utility query we can can get useful and UN non-disclosing views whenever this is not ruled out for trivial reasons:

Theorem 6. *[UN non-disclosure minimal d-views in the presence of replication] If BCQ Q contains a relation of non-zero arity replicated across all sources then there is a d-view that is useful for Q and UN non-disclosing for BCQ p if and only if there is no homomorphism of p to Q. Further we can use the same d-view for every such p without a homomorphism into a given Q. In particular, there is a view that is UN non-disclosure minimal for Q.*

Thus even though we do not have minimally informative useful d-views, we have d-views that are optimal from the perspective of UN non-disclosure and utility for a fixed Q.

We sketch the idea of the proof of Theorem 6. Given utility query Q and local instance $\mathcal{I}$ we can form the "product instance" of $\mathcal{I}$ and Q. The elements of a product instance will be pairs (x, c) where x is a variable of Q and c an element of $\mathcal{I}$, and there will be atom $R((x_1, c_1), \ldots, (x_n, c_n))$ in the product exactly when there are corresponding atoms in $\mathsf{CanView}^s(Q)$ and $\mathcal{I}$. Thus we will have homomorphisms from this instance to both $\mathcal{I}$ and $\mathsf{CanView}^s(Q)$. We use ECRs that make an s-instance $\mathcal{I}$ equivalent to all instances formed by iterating this product construction of $\mathcal{I}$ with $\mathsf{CanView}^s(Q)$. The d-views corresponding to these ECRs will be UN non-disclosing because in the product we will have a fresh copy of any partial match of Q, and so the only way for the secret query p to hold in the product will be if it has a homomorphism into Q, which is forbidden by hypothesis. The replication constraint ensures that if we have two d-instances that are equivalent, where the interpretation of the replicated relation is non-empty, the number of iterations of the product construction is the same on each source. Using this fact we can ensure that the d-views are useful.

Example 4. *We give an example of the power of Theorem 6, and we highlight the difference from the situation with only local constraints. Suppose we have two sources, with one binary relation S replicated between the two, and each source having one non-replicated binary relation, R in one source and T in the other. The utility query Q is $\exists x, y.R(x, y) \wedge S(x, y) \wedge T(x, y)$ and the secret query p is $\exists x.R(x, x)$.*

Since p is not entailed by Q, Theorem 6 implies that there are views that are useful for Q but UN non-disclosing for p. But it is easy to see that the canonical d-view of Q is UN disclosing for p.

The views used in Theorem 6 are not isomorphism-invariant: like the views from Proposition 6, the product construction can be seen as applying some value transformation on the elements of each instance. We can show — in sharp contrast to the situation with only local constraints — that with replication, even to achieve this weaker notion of minimality, it may be essential to use d-views based on queries that are not isomorphism-invariant.

Proposition 7. *There is a d-schema with replication, and BCQs Q and p such that there is a d-view useful for Q and UN non-disclosing for p, but there is no such d-view based on queries returning values in the active domain and commuting with isomorphisms. In particular, we cannot find a UN non-disclosure minimal d-view that is isomorphism-invariant in the above sense.*

6 Discussion and Outlook

We have studied the ability to design views that satisfy diverse goals: expressiveness requirements in terms of full disclosure of a specified set of queries in the context of data integration, and inexpressiveness restrictions in terms of either minimal utility or minimizing disclosure of queries. Our main results characterize information-theoretically minimal views that support the querying of a given CQ.

We consider only a limited setting; e.g. CQs for the utility query. Our hope is that the work can serve as a basis for further exploration of the trade-offs in using query-based mechanisms in a variety of settings.

Even in this restricted setting, our contribution focuses primarily on expressiveness, leaving open many questions of decidability and complexity. In particular we do not know whether the Σ_2^p bound of Corollary 2 is tight. Nor do we know whether the analogous question for arbitrary views – whether there is an arbitrary d-view that is useful for a given Q but UN non-disclosing for a given p – is even decidable. Our results reduce this to a non-disclosure question for the shuffle views.

Lastly, we mention that our positive results about CQ views (e.g. in Section 3) rely on an analysis of the chase, which can be infinite. Thus they are only proven for a semantics of usefulness that considers all instances. We believe that the analogous results where only finite instances are considered can easily be proven using the techniques of Section 4, but leave this for future work.

Acknowledgements

Benedikt was supported by EPSRC grant EP/M005852/1.

References

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

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.

Baget, J.-F.; Mugnier, M.-L.; Rudolph, S.; and Thomazo, M. 2011. Walking the complexity lines for generalized guarded existential rules. In *IJCAI*.

Baget, J.; Garreau, F.; Mugnier, M.; and Rocher, S. 2014. Extending acyclicity notions for existential rules. In *ECAI*.

Benedikt, M.; Bourhis, P.; ten Cate, B.; and Puppis, G. 2016. Querying visible and invisible information. In *LICS*.

Benedikt, M.; Bourhis, P.; Jachiet, L.; and Thomazo, M. 2019. Reasoning about disclosure in data integration in the presence of source constraints. In *IJCAI*.

Benedikt, M.; Cuenca Grau, B.; and Kostylev, E. V. 2018. Logical foundations of information disclosure in ontology-based data integration. *AI* 262:52–95.

Benedikt, M.; ten Cate, B.; and Tsamoura, E. 2016. Generating plans from proofs. In *TODS*.

Bonatti, P. A., and Sauro, L. 2013. A confidentiality model for ontologies. In *ISWC*.

Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Rosati, R. 2012. View-based Query Answering in Description Logics: Semantics and Complexity. *J. Comput. Syst. Sci.* 78(1):26–46.

Chaum, D.; Crépeau, C.; and Damgard, I. 1988. Multiparty unconditionally secure protocols. In *STOC*.

Cuenca Grau, B.; Horrocks, I.; Krötzsch, M.; Kupke, C.; Magka, D.; Motik, B.; and Wang, Z. 2013. Acyclicity notions for existential rules and their application to query answering in ontologies. *JAIR* 47:741–808.

Dwork, C., and Roth, A. 2014. The algorithmic foundations of differential privacy. *Found. & Trends in Th. Comp. Sci.* 9(3&4):211–407.

Dwork, C. 2006. Differential privacy. In *ICALP*.

Gogacz, T., and Marcinkowski, J. 2015. The hunt for a red spider: Conjunctive query determinacy is undecidable. In *LICS*.

Gogacz, T., and Marcinkowski, J. 2016. Red spider meets a rainworm: Conjunctive query finite determinacy is undecidable. In *PODS*.

Halevy, A. Y. 2001. Answering queries using views: A survey. *VLDB J.* 10(4):270–294.

Li, C.; Li, D. Y.; Miklau, G.; and Suciu, D. 2017. A theory of pricing private data. *CACM* 60(12):79–86.

Marnette, B. 2009. Generalized schema-mappings: from termination to tractability. In *PODS*.

Nash, A., and Deutsch, A. 2007. Privacy in GLAV information integration. In *ICDT*.

Nash, A.; Segoufin, L.; and Vianu, V. 2010. Views and queries: Determinacy and rewriting. *TODS* 35(3).