Logical Foundations of Information Disclosure in Ontology-Based Data Integration

Michael Benedikt, Bernardo Cuenca Grau, Egor V. Kostylev

Department of Computer Science, University of Oxford, UK

Abstract

Ontology-based data integration systems allow users to effectively access data sitting in multiple sources by means of queries over a global schema described by an ontology. In practice, data sources often contain sensitive information that the data owners want to keep inaccessible to users. Our aim in this paper is to lay the logical foundations of information disclosure in ontology-based data integration. Our focus is on the semantic requirements that a data integration system should satisfy before it is made available to users for querying, as well as on the computational complexity of checking whether such requirements are fulfilled. In particular, we formalise and study the problem of determining whether a given data integration system discloses a source query to an attacker. We consider disclosure on a particular dataset, and also whether a schema admits a dataset on which disclosure occurs. We provide matching lower and upper complexity bounds on disclosure analysis, in the process introducing a number of techniques for analysing logical privacy issues in ontology-based data integration.

Keywords: Knowledge Representation and Reasoning, Ontologies, Ontology-based Data Access, Data Integration, Query Answering, Data Privacy.

1. Introduction

Data integration systems expose information from multiple, heterogeneous data sources by means of a *global schema*, in which the mismatches between the individual schemata of the data sources have been reconciled [35, 43, 45, 56, 76]. In addition to reconciling the structure of the data sources, the global schema also enables uniform access to the data by providing users with the vocabulary for query formulation. The relationships between the data sources and the global schema are determined by *mappings*—that is, logical formulae that declaratively specify how each term in the global schema relates to the data in the sources. Queries issued against the global schema are typically answered by one of two approaches. In the first approach, an instance over the global schema is initially materialised using the mappings and the data in the sources; then, the query is answered over the materialised instance. In the second approach, no data is exported from the sources and the global schema remains virtual; this is achieved by first reformulating the user query on-the-fly into a set of queries over the sources, and then assembling back their results.

In *ontology-based data integration* the global schema is extended to an *ontology*—that is, a first-order logic theory expressed in an ontology language such as the Web Ontology Language (OWL) [29]. In addition to defining a high-level conceptual view over the data, the formulae in the ontology also specify how terms in the vocabulary relate to each other, thus providing valuable domain background knowledge that can be exploited to enrich query answers with implicit information.

An important instantiation of the ontology-based data integration paradigm is *ontology-based data access (OBDA)* [23, 68], where the ontology and mapping languages are typically restricted so as to ensure the *first-order rewritability* property: given an arbitrary conjunctive query q over the global schema, ontology $\mathcal{O}$ and mappings $\mathcal{M}$ in the relevant languages, it is possible to reformulate q into a first-order query q' over the source schema such that, for any source data instance $\mathcal{D}$, the answers to q over $\mathcal{O}$, $\mathcal{M}$ and $\mathcal{D}$ coincide with those to q' over $\mathcal{D}$ alone. First-order rewritability therefore ensures that queries can be answered following the second, virtual approach, where the ontology axioms must now also be taken into account during query reformulation. Ontology languages with this property include the QL profile of OWL 2 [65], which is based on the Description Logic DL-Lite$_{\mathcal{R}}$ [22], as well as first-order rule formalisms such as linear tuple-generating dependencies (TGDs) [19, 47]. OBDA has received in recent years a great deal of attention in all its dimensions, including theoretical research [3, 4, 11, 12, 23, 40, 50, 55], system implementation [20, 21, 46], and industrial applications [25, 38, 44, 48, 49].

An important aspect of ontology-based data integration that has so far received only relatively little attention is that of preventing unauthorised information disclosure [24, 26, 31, 66]. In practice, data sources often contain sensitive information, and it is well-known that information integration and linkage poses major threats to the confidentiality of such sensitive data

Email addresses: michael.benedikt@cs.ox.ac.uk (Michael Benedikt), bernardo.cuenca.grau@cs.ox.ac.uk (Bernardo Cuenca Grau), egor.kostylev@cs.ox.ac.uk (Egor V. Kostylev)

even if it is only made available in an anonymised form [75]. In the context of ontology-based data integration, the risks of unauthorised information disclosure quickly become apparent since the information exposed to users depends on a complex combination of schema reconciliation, reasoning over the ontology, and access to data in the sources via the mappings.

Our aim in this paper is to lay the logical foundations of information disclosure in ontology-based data integration. Our focus is on the semantic requirements that a data integration system should satisfy before it is made available to users for querying, as well as on the computational complexity of checking whether such requirements are fulfilled. These are fundamental initial steps towards the development and implementation of algorithms suitable for applications.

In Section 3, we present a general logical framework for information disclosure that builds upon the work by Nash and Deutsch [66], Chirkova and Yu [26], and Benedikt et al. [7]. In line with existing logic-based approaches to data privacy and confidentiality enforcement in databases [7, 10, 26, 64, 66] and knowledge representation [16, 27, 31, 32, 34, 73], our framework assumes that sensitive information is represented by a query (the *policy*) over the source schema, and also that all schema-level information (ontology, mappings, source schema, and policy specification) is publicly available—this is a worst-case scenario for confidentiality enforcement. In contrast, the actual data is not made available directly, but rather only by means of queries over the global schema.

In this setting, the information that users can gather about the actual source data is inherently incomplete: different source data instances may be *indistinguishable*, in the sense that users cannot tell the difference between them by querying the system. In Section 3.1 we define source instances $\mathcal{D}$ and $\mathcal{D}'$ to be indistinguishable for ontology $\mathcal{O}$ and mappings $\mathcal{M}$ if, for every query q over the global schema, the answers to q over $\mathcal{O}$, $\mathcal{M}$, and $\mathcal{D}$ coincide with the answers to q over $\mathcal{O}$, $\mathcal{M}$ and $\mathcal{D}'$. In addition to playing a fundamental role in our framework, source indistinguishability is also an interesting notion in its own right. On the one hand, it has obvious applications in practice: for example, it can be used to determine whether given changes in the data can affect applications querying the system; on the other hand, indistinguishability notions akin to ours have received significant attention in the ontology literature in recent years [11, 18, 51].

Disclosure in our framework occurs when users are able to uncover an answer to the policy query over the source data by querying the system and exploiting the availability of schema-level information (e.g., they are able to uncover that a specific hospital patient is an oncology patient, where the policy is a query asking for the list of all oncology patients). In particular, this happens when the policy answer holds in all the (possibly infinitely many) source instances indistinguishable from the real source instance. Indeed, in such a situation an attacker can in principle uncover the policy answer with certainty. If no such disclosure is possible, we say in Section 3.2 that the system *complies* to the policy.

In order to meet the requirements of applications with stricter privacy requirements, we additionally introduce in Section 3.3 a stronger notion of compliance, where we additionally require that a user cannot uncover non-answers to the policy (e.g., to uncover that a specific hospital patient is *not* an oncology patient). Such negative information can be valuable to an attacker; for instance, by ruling out specific hospital patients as oncology patients, one may increase the likelihood that the remaining patients are indeed oncology patients. If a data integration system satisfies such stricter requirement, we say that it *strongly complies* to the policy.

Our framework also specifies in Section 3.4 the natural data-independent variants of the aforementioned notions, where compliance must hold regardless of the specific data stored in the sources. Having guarantees that depend only on the (relatively static) definition of the ontology and mappings, but not on the actual contents of the data sources can be important in practice since data updates are frequent operations in many applications.

Finally, Section 3.5 discusses the specific role that the ontology plays in information disclosure, and in particular on source indistinguishability and policy (strong) compliance.

The core technical sections of our paper, the results of which are summarised in Section 4, are devoted to an in-depth study of the decision problems associated to source indistinguishability and compliance checking in all its aforementioned variants. In our analysis, we consider arbitrary first-order ontology languages and parameterise our main results in terms of the complexity for standard query answering for these languages (e.g., NP-complete for DL-Lite$_{\mathcal{R}}$). Similarly, we consider the general case of GLAV (global-local-as-view) mappings as well as its well-known special cases GAV (global-as-view), CQ views, and LAV (local-as-view) [56]. In all cases, we put special emphasis on the results most relevant to standard OBDA settings where the ontology is expressed in DL-Lite$_{\mathcal{R}}$ and the mappings are GAV.

In Section 5 we study the source indistinguishability problem and establish tight complexity bounds for a wide range of ontology and mapping languages. We show that source indistinguishability is often no harder than standard query entailment; however, in many other cases of practical relevance (including standard OBDA) indistinguishability checking is strictly harder than query entailment under standard complexity-theoretic assumptions.

In Section 6, we study the computational properties of policy compliance and its stronger variant as defined in Sections 3.2 and 3.3. In Section 6.1 we propose a generic non-deterministic algorithm for (strong) compliance checking. The correctness of our algorithm shows that both versions of compliance are decidable whenever the ontology language of choice has a decidable query entailment problem. Furthermore, the analysis of our algorithm provides upper bounds for many of the most relevant cases, ranging from NP to NExpTime, where the complexity for OBDA with DL-Lite$_{\mathcal{R}}$ ontologies and GAV mappings lies in the second level of the polynomial hierarchy in general and in NP in data complexity. In Section 6.2 we show that all the

upper bounds provided by our generic compliance checking algorithm are tight, where many of our matching lower bounds hold already under the assumption that the ontology is empty. Finally, we address the practical concerns raised by the high complexity of compliance checking by identifying practically relevant islands of tractability.

In Section 7, we study the data-independent notions of compliance defined in Section 3.4 and show that the problem is already undecidable for OBDA, even under the assumption that the ontology is empty. We then isolate a decidable case and study an additional restriction ensuring tractability.

Our results have significant implications on related work, which are discussed in detail in Sections 8 and 9. On the one hand, they correct some of the complexity bounds claimed by Nash and Deutsch [66]; on the other hand, our work also closes open problems in *data pricing*—the problem of automatically assigning a fair price to a chunk of data given the price of a set of views [54]. In particular, we show that our results on policy compliance immediately imply a Π_2^p lower bound to the so-called *instance-based determinacy* problem, which is at the core of the pricing framework by Koutris et al. [54].

Some of the results of this paper appeared in an earlier conference publication [8].

2. Preliminaries

In this section, we recapitulate the definitions of standard computational complexity classes and logical formalisms with their associated reasoning problems that we use in the rest of the paper.

2.1. Complexity Classes

We use the standard definitions of the basic time complexity classes such as P, NP, coNP, ExpTime and NExpTime, as well as of the basic space complexity classes such as PSpace, LogSpace, and NLogSpace [67]. For complexity classes C and C', we denote with $C^{C'}$ the class of decision problems that can be solved by a Turing machine running in C and using an oracle for decision problems in C'. The *polynomial hierarchy* is then defined inductively as follows:

$$\Sigma_0^P = \Pi_0^P = \Delta_0^P = \mathrm{P}, \qquad \Sigma_{k+1}^P = \mathrm{NP}^{\Sigma_k^P}, \qquad \Pi_{k+1}^P = \mathrm{coNP}^{\Sigma_k^P}, \quad \text{and} \quad \Delta_{k+1}^P = \mathrm{P}^{\Sigma_k^P}.$$

We also consider $\mathrm{P}^{\|\mathrm{NP}}$—the class of problems solvable in P with nonadaptive calls to an NP oracle [77]. Note the difference between $\mathrm{P}^{\|\mathrm{NP}}$ and Δ_2^P, where the input of an oracle call can depend on the result of previous calls.

Finally, we consider the class AC^0 used in circuit complexity, which encompasses all families of circuits of constant depth and polynomial size with unlimited fan-in AND and OR gates (e.g., see [67]). The AC^0 class is strictly contained in LogSpace and is especially important in our setting as it corresponds to the data complexity of query evaluation in databases.

2.2. Tuple-Generating Dependencies, Ontologies, and Queries

We adopt standard notions from function-free first-order logic over a vocabulary of relational names with fixed arities and constants. A *fact* is a ground atom, and an *instance* is a finite set of facts. We denote with $\mathit{ADom}(\mathcal{D})$ the set of constants occurring in an instance $\mathcal{D}$.

A *tuple-generating dependency (TGD)* is an implicitly universally quantified sentence of the form $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ are disjoint tuples of distinct variables, while the *body* $\varphi(\mathbf{x}, \mathbf{z})$ and the *head* $\psi(\mathbf{x}, \mathbf{y})$ are conjunctions of atoms such that each term is either a constant or a variable in $\mathbf{x} \cup \mathbf{z}$ and $\mathbf{x} \cup \mathbf{y}$, respectively, and $\varphi(\mathbf{x}, \mathbf{z})$ mentions all variables $\mathbf{x}$. Variables $\mathbf{x}$, common for the head and the body, are called the *frontier variables*. A TGD is *full* if it does not have existentially quantified variables and the head consists of a single atom; it is *linear* if its body consists of a single atom.

In this paper, we adopt a very general notion of an *ontology*, which is not constrained to specific fragments of first-order logic, such as description logics. In particular, we often refer to an *ontology language* $\mathbb{O}$ to denote any first-order fragment, and define an ontology in $\mathbb{O}$ as a finite set of first-order sentences belonging to $\mathbb{O}$. Many of our complexity results, however, are specific to ontologies consisting of TGDs of a specific form. In such cases, the relevant restrictions will be pointed out explicitly; for example, an ontology is *full* (or *linear*) if it consists of full (or linear, respectively) TGDs, and an ontology language is *full* (or *linear*) if it consists of full (or linear, respectively) ontologies. Another ontology language that is widely used in practice in the context of data integration is the description logic *DL-Lite*$_{\mathcal{R}}$ [23], which can be seen as the fragment consisting of linear TGDs of the following forms, where A and B are unary relational names while R and S are binary relational names:

$$A(x) \rightarrow B(x), \quad A(x) \rightarrow \exists y.\, R(x, y), \quad R(x, y) \rightarrow A(x), \quad R(x, y) \rightarrow A(y), \quad R(x, y) \rightarrow S(x, y), \quad \text{and} \quad R(x, y) \rightarrow S(y, x).$$

Finally, some of our results are formulated for the trivial ontology language that contains only the empty ontology.

A *conjunctive query* (*CQ*) $q(\mathbf{x})$ with *free* variables $\mathbf{x}$ is a formula $\exists \mathbf{y}.\, \varphi(\mathbf{x}, \mathbf{y})$, where $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunction of atoms such that it mentions all variables $\mathbf{x}$ and each term is a constant or a variable in $\mathbf{x} \cup \mathbf{y}$. Note that the head of a TGD is always a CQ, so we may sometimes refer to it as the *head CQ*. A *union* $q(\mathbf{x})$ of CQs (*UCQ*) with *free* variables $\mathbf{x}$ is a disjunction of CQs with free variables $\mathbf{x}$. The *arity* of a (U)CQ $q(\mathbf{x})$ is the number of variables in $\mathbf{x}$; (U)CQs of arity 0 are *Boolean*.

The basic reasoning problem we consider is CQ entailment for an ontology language $\mathbb{O}$: a Boolean CQ q is *entailed* by an ontology $\mathcal{O}$ in $\mathbb{O}$ and an instance $\mathcal{D}$ if and only if $\mathcal{O} \cup \mathcal{D} \models q$. *Fact entailment* is the restriction of CQ entailment where the input CQ is a fact. It is well-known that both CQ and fact entailment are undecidable for the language of all TGDs, whereas both problems become ExpTime- and PSpace-complete for full and linear TGDs, respectively [1, 47]. Finally, CQ entailment for DL-Lite$_\mathcal{R}$ is NP-complete, whereas fact entailment is NLogSpace-complete and thus tractable [23].

We also consider *data complexity* of CQ and fact entailment—that is, the complexity of the same problems under the assumption that only $\mathcal{D}$ constitutes the input, while $\mathcal{O}$ and q are fixed. It is known that both problems are undecidable even in data complexity for arbitrary TGDs, while they are P-complete and in AC^0 in data complexity for full TGDs and linear TGDs (and hence DL-Lite$_\mathcal{R}$), respectively. Sometimes we require, for complexity classes $\mathcal{C}$ and $\mathcal{C}'$, CQ (or fact) entailment to be in $\mathcal{C}$ with $\mathcal{C}'$ data complexity: by this we mean that there should exist an algorithm witnessing both bounds.

2.3. Ontology-Based Data Integration

Following [56], we assume that the set of the relational names in a vocabulary is split into two disjoint subsets: *source* and *global schema*. The *arity* of such a schema is the maximal arity of its relational names.

A *GLAV (global-local-as-view) mapping*, or simply *mapping*, for short, is a TGD with the body mentioning relational names only in the source schema and the head mentioning relational names only in the global schema. Following a common convention, we call full mappings *GAV* (*global-as-view*) and linear mappings *LAV* (*local-as-view*). Note that these notions are incomparable, and a mapping can be both GAV and LAV at the same time.

A set of mappings $\mathcal{M}$ is *GAV* (or *LAV*) if so are all the mappings in $\mathcal{M}$. The *frontier size* of $\mathcal{M}$ is the maximum number of frontier variables in a mapping from $\mathcal{M}$. Finally, $\mathcal{M}$ is a set of *CQ views* if it is GAV and no two mappings in $\mathcal{M}$ share the same head relation name.

A *mapping language* $\mathbb{M}$ consists of finite sets of mappings. It is *GAV* (or *CQ views*, or *LAV*) if so are all $\mathcal{M}$ in $\mathbb{M}$. Its source (or global) arity is *bounded by* a fixed integer k if the arity of the source (or global, respectively) schema of each $\mathcal{M} \in \mathbb{M}$ is at most k, and it is of *bounded source* (or *global*, respectively) arity if there exists such an integer k. Similarly, its frontier is bounded by a fixed integer k if the frontier size is at most k, and it is of *bounded frontier* if there exists such an integer k. Note that, for GAV (and, hence, CQ views) mapping languages, bounded global arity implies bounded frontier; similarly, for LAV mapping languages, bounded source arity implies bounded frontier.

A *data integration setting* is a triple $(\mathcal{O}, \mathcal{M}, \mathcal{D})$, where $\mathcal{O}$ is an ontology over the global schema, $\mathcal{M}$ is a finite set of mappings, and $\mathcal{D}$ is an instance over the source schema. The standard setting in *ontology-based data access* (OBDA) is when $\mathcal{O}$ is restricted to be a DL-Lite$_\mathcal{R}$ ontology and $\mathcal{M}$ to be GAV; in this case, the frontier and global arity are automatically bounded by 2.

An interpretation I over the global schema is a *model* of a data integration setting $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ if $I \models \mathcal{O}$ and, for each mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and each tuple of constants $\mathbf{a}$, it holds that $I \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ whenever $\mathcal{D} \models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$. For $q(\mathbf{x})$ a CQ over the global schema, a tuple $\mathbf{a}$ of constants is a *certain answer* to $q(\mathbf{x})$ over $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ if $I \models q(\mathbf{a})$ for all models I of $(\mathcal{O}, \mathcal{M}, \mathcal{D})$.

The *virtual image* $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ of a set of mappings $\mathcal{M}$ and an instance $\mathcal{D}$ is the following set of Boolean CQs:

$$\{\, \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y}) \mid \varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y}) \text{ in } \mathcal{M},\ \mathbf{a} \text{ a tuple of constants, and } \mathcal{D} \models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z}) \,\}.$$

The following simple proposition establishes the connection between computing certain answers in data integration and standard CQ entailment in first-order logic.

Proposition 1. *A tuple* $\mathbf{a}$ *of constants is a certain answer to a CQ* $q(\mathbf{x})$ *over the global schema over a data integration setting* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ *if and only if* $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models q(\mathbf{a})$.

Proof. It suffices to show that an interpretation I is a model of $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ if and only if $I \models \mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$. Assume that $I \models \mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and let $\mathbf{a}$ be a tuple of constants satisfying $\mathcal{D} \models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$ for a mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$. Since $I \models \mathcal{V}_{\mathcal{M},\mathcal{D}}$, we have that $I \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ and hence, since $\mathbf{a}$ and the mapping are arbitrary as well as $I \models \mathcal{O}$, interpretation I is a model of $(\mathcal{O}, \mathcal{M}, \mathcal{D})$. Conversely, if I is a model of $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ then $I \models \mathcal{O}$. To see that $I \models \mathcal{V}_{\mathcal{M},\mathcal{D}}$, let $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ be an arbitrary Boolean CQ in $\mathcal{V}_{\mathcal{M},\mathcal{D}}$; by construction of $\mathcal{V}_{\mathcal{M},\mathcal{D}}$, we have that $\mathcal{D} \models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$ for a mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$. But then, the fact that I is a model of $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ ensures that $I \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$, as required. □

3. Framework for Information Disclosure

In this section, we present our framework for information disclosure and define the associated reasoning problems. All the definitions are accompanied by simple motivating examples.

To start with, let us consider the information system of a hospital where data about patient appointments is stored in various databases. Although various departments store appointment data using different schemata, they all share basic attributes, such as

the IDs for patients, and doctors, as well as the appointment dates.[1] For instance, the oncology department database contains a relation

$$\mathsf{OncologyAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Treatment}),$$

where *PatientID* and *DoctorID* represent patient and doctor IDs, *Date* represents the date of the appointment, and *Treatment* represents the prescribed treatment. The paediatrics department may use a database containing a relation

$$\mathsf{PaediatricsAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Location}),$$

where *Location* indicates the room in which the appointment took place.

To integrate this data, the hospital relies on a global schema capturing the common terminology in all appointments. This global schema includes relational names such as $\mathsf{Appointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date})$, $\mathsf{OncologistRecord}(\mathit{DoctorID}, \mathit{Date})$, and $\mathsf{PaediatricianRecord}(\mathit{DoctorID}, \mathit{Date})$. The following simple mappings $\mathcal{M}^{ex}$, which are both GAV and LAV, relate the source schema to the global schema:

$$\mathsf{OncologyAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Treatment}) \rightarrow \mathsf{Appointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}), \tag{1}$$

$$\mathsf{OncologyAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Treatment}) \rightarrow \mathsf{OncologistRecord}(\mathit{DoctorID}, \mathit{Date}), \tag{2}$$

$$\mathsf{PaediatricsAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Location}) \rightarrow \mathsf{Appointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}), \tag{3}$$

$$\mathsf{PaediatricsAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Location}) \rightarrow \mathsf{PaediatricianRecord}(\mathit{DoctorID}, \mathit{Date}). \tag{4}$$

The data integration system may also come with an ontology, which establishes the relationships between the relational names in the global schema. For instance, the following ontology $\mathcal{O}^{ex}$ establishes that oncologist and paediatrician records are doctor records, and that a person mentioned in a doctor's record has a medical degree:

$$\mathsf{OncologistRecord}(\mathit{DoctorID}, \mathit{Date}) \rightarrow \mathsf{DoctorRecord}(\mathit{DoctorID}, \mathit{Date}), \tag{5}$$

$$\mathsf{PaediatricianRecord}(\mathit{DoctorID}, \mathit{Date}) \rightarrow \mathsf{DoctorRecord}(\mathit{DoctorID}, \mathit{Date}), \tag{6}$$

$$\mathsf{DoctorRecord}(\mathit{DoctorID}, \mathit{Date}) \rightarrow \exists \mathit{qID}.\, \mathsf{hasQualification}(\mathit{DoctorID}, \mathit{qID}) \wedge \mathsf{MedDegree}(\mathit{qID}). \tag{7}$$

The schema designers may not want to disclose the relationship between patients and the departments the patients have visited. However, the confidentiality of such information is at risk. For example, consider the following simple instance $\mathcal{D}_1^{ex}$ providing information about two oncology appointments and two paediatrics appointments as per the aforementioned schemata.

$\mathcal{D}_1^{ex}$:

OncologyAppointment	*PatientID*	*DoctorID*	*Date*	*Treatment*
	*P*1	*D*1	30-11-2017	Daunorubicin
	*P*2	*D*2	10-06-2017	Docetaxel

PaediatricsAppointment	*PatientID*	*DoctorID*	*Date*	*Location*
	*P*3	*D*3	15-10-2017	Room 225
	*P*4	*D*4	10-08-2017	Room 230

The accessible information is determined by the virtual image of $\mathcal{D}_1^{ex}$, which instantiates the global relations mentioned in the mappings as given next.

$\mathcal{V}_{\mathcal{M},\mathcal{D}_1^{ex}}$:

Appointment	*PatientID*	*DoctorID*	*Date*
	*P*1	*D*1	30-11-2017
	*P*2	*D*2	10-06-2017
	*P*3	*D*3	15-10-2017
	*P*4	*D*4	10-08-2017

OncologistRecord	*DoctorID*	*Date*
	*D*1	30-11-2017
	*D*2	10-06-2017

PaediatricianRecord	*DoctorID*	*Date*
	*D*3	15-10-2017
	*D*4	10-08-2017

By querying OncologistRecord, an attacker can determine that doctors *D*1 and *D*2 had oncology appointments on the 30th of November and the 10th of June, respectively. Furthermore, from Appointment, the attacker has access to a list of all the appointments a doctor had on a given date; in particular, patients *P*1 and *P*2 were the only ones to have appointments with

[1] In practice, such common attributes typically have different names and their identification is the result of a prior schema matching process [35].

doctors $D1$ and $D2$, respectively, on the aforementioned dates. As a result, the attacker can infer that patients $P1$ and $P2$ had oncology appointments.

In this case, the unauthorised disclosure depends on the attacker's ability to "trace back" using the mappings the exact relation in the source that exported each tuple in the extension of the relevant global relations. In particular, the definition of the mappings is assumed to be available; thus, an attacker knows, on the one hand, that the list of appointments consists of the union of the oncology and paediatrics appointment data at the source and, on the other hand, that the oncologist records are solely derived from the oncology appointment data.

In what follows, we will make these intuitions precise.

3.1. Source Indistinguishability

In a data integration setting, users (including malicious attackers) can only interact with the system by posing queries against the global schema. Users have no direct access to the source instances and hence the information they can gather about the source data is inherently incomplete. As a result of this incompleteness, different source instances may be *indistinguishable*, in the sense that users cannot tell the difference between them by querying the system. In our motivating example, users cannot access any information about prescribed treatments or appointment locations in the source relations since these attributes are not exposed by the mappings. As a result, the following source instance $\mathcal{D}_2^{ex}$ cannot be told apart from $\mathcal{D}_1^{ex}$ by means of user queries against the global schema; in particular we can observe that their virtual images $\mathcal{V}_{\mathcal{M},\mathcal{D}_1^{ex}}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}_2^{ex}}$ coincide.

$\mathcal{D}_2^{ex}$:

OncologyAppointment	*PatientID*	*DoctorID*	*Date*	*Treatment*
	*P*1	*D*1	30-11-2017	Abemaciclib
	*P*2	*D*2	10-06-2017	Flutamide

PaediatricsAppointment	*PatientID*	*DoctorID*	*Date*	*Location*
	*P*3	*D*3	15-10-2017	Room 305
	*P*4	*D*4	10-08-2017	Room 400

Definition 2. *Source instances $\mathcal{D}$ and $\mathcal{D}'$ are* indistinguishable *for an ontology $\mathcal{O}$ and mappings $\mathcal{M}$ if, for every CQ q over the global schema, the certain answers to q over $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ and $(\mathcal{O}, \mathcal{M}, \mathcal{D}')$ coincide.*

The associated decision problem for an ontology language $\mathbb{O}$ and mapping language $\mathbb{M}$ is defined as follows, where $\mathcal{O}$ ranges over $\mathbb{O}$ and $\mathcal{M}$ over $\mathbb{M}$: SourceInd$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *is* `true` *if and only if $\mathcal{D}$ and $\mathcal{D}'$ are indistinguishable for $\mathcal{O}$ and $\mathcal{M}$. We consider not only the combined complexity of* SourceInd$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$*, but also its* data complexity*—that is, the complexity under assumption that $\mathcal{O}$ and $\mathcal{M}$ are fixed, while only $\mathcal{D}$ and $\mathcal{D}'$ form the input.*

The notion of indistinguishability in Definition 2 can be used to formalise information disclosure. Intuitively, by knowing $\mathcal{O}$ and $\mathcal{M}$, all a malicious attacker querying the system can gather from the source instance $\mathcal{D}$ is that it must be one of the (possibly infinitely many) source instances $\mathcal{D}'$ indistinguishable from $\mathcal{D}$. We next provide a fundamental characterisation of source indistinguishability, which we exploit extensively throughout this paper. Our characterisation can be seen as an extension of Theorem 1 of [66] to the setting with an ontology.

Lemma 3. *The following are equivalent for any ontology $\mathcal{O}$, mappings $\mathcal{M}$, and source instances $\mathcal{D}$ and $\mathcal{D}'$:*

1. SourceInd$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *is* `true`*;*
2. *for each mapping in $\mathcal{M}$ with head CQ q, the certain answers to q over $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ and $(\mathcal{O}, \mathcal{M}, \mathcal{D}')$ coincide; and*
3. *$\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ are logically equivalent.*

Proof. *Statement 1* implies *Statement 2* by definition. Furthermore, *Statement 2* implies *Statement 3* by Proposition 1. We next show that *Statement 3* implies *Statement 1*. Assume that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ are equivalent. Let $q(\mathbf{x})$ be a CQ over the global schema, and let $\mathbf{a}$ be a certain answer to $q(\mathbf{x})$ over $(\mathcal{O}, \mathcal{M}, \mathcal{D})$. We show that the same holds over $(\mathcal{O}, \mathcal{M}, \mathcal{D}')$. Assume to the contrary that this is not the case. Then, there exists an interpretation I' over both source and global schemata that satisfies both $\mathcal{M}$ and $\mathcal{O}$, has source part corresponding precisely to $\mathcal{D}'$, and does not satisfy $q(\mathbf{a})$. Let I be an interpretation formed by replacing the interpretations of the source relational names in I' by the part corresponding to $\mathcal{D}$. Clearly, I satisfies $\mathcal{O} \wedge \neg q(\mathbf{a})$ since these mention only the global relational names, which are unchanged from I'. Also, the source part of I corresponds to $\mathcal{D}$ by definition. To see that I satisfies $\mathcal{M}$, consider any homomorphism h from the body of a mapping into $\mathcal{D}$. The corresponding image of the mapping's head is in $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and thus, by hypothesis, it is implied by $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$. However, I' satisfies $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$, and thus I' satisfies the image of the head as well. Since I agrees with I' on the global relations, the desired contradiction follows. Arguing symmetrically, we see that a certain answer of q over $(\mathcal{O}, \mathcal{M}, \mathcal{D}')$ is a certain answer of q over $(\mathcal{O}, \mathcal{M}, \mathcal{D})$. □

The second statement in the lemma shows that, to check indistinguishability, we can restrict ourselves to consider only those CQs that appear in the head of the mappings (rather than the infinitely many possible queries that can be formed using the global vocabulary). The third statement shows that indistinguishability checking amounts to solving a logical entailment problem involving the ontology and the virtual images of the relevant source instances over the mappings.

3.2. Policies, Information Disclosure, and Policy Compliance

The sensitive information in a data integration setting can be declaratively represented by a query over the source schema, which we refer to as the *policy*.

Definition 4. *A* policy *is a CQ over the source schema. A* policy language $\mathbb{P}$ *is a set of policies. It is of* arity bounded *by a fixed integer m if every CQ in* $\mathbb{P}$ *has at most m free variables. It is of* bounded arity *if it is of arity bounded by some m. It is* ground *if it consists of conjunctions of facts.*

In our motivating example, the requirement that the identity of patients with an oncology appointment cannot be disclosed can be represented by the following policy:

$$p^{ex}(\mathit{PatientID}) = \exists \mathit{DoctorID}, \mathit{Date}, \mathit{Treatment}.\, \mathsf{OncologyAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Treatment}).$$

Intuitively, disclosure of sensitive information occurs whenever there is a certain answer to the policy that holds in all the source instances that are indistinguishable from the point of view of the attacker. In such a situation the attacker would be able to uncover the aforementioned answer. If no such disclosure can occur, we say that the data integration setting *complies* to the policy.

Definition 5. *A data integration setting* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ complies to *a Boolean policy p if there exists a source instance* $\mathcal{D}'$ *indistinguishable from* $\mathcal{D}$ *for* $\mathcal{O}$ *and* $\mathcal{M}$ *such that* $\mathcal{D} \models p$ *implies* $\mathcal{D}' \not\models p$*. Furthermore,* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ complies *to a policy* $p(\mathbf{x})$ *with m free variables* $\mathbf{x}$ *if it complies to* $p(\mathbf{a})$ *for all m-tuples* $\mathbf{a}$ *of constants from* $\mathcal{D}$*.*

The associated decision problem for an ontology language $\mathbb{O}$*, mapping language* $\mathbb{M}$*, and policy language* $\mathbb{P}$ *is defined as follows, where* $\mathcal{O}$ *ranges over* $\mathbb{O}$*,* $\mathcal{M}$ *over* $\mathbb{M}$*, and p over* $\mathbb{P}$*:* COMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *is* `true` *if and only if* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ *complies to p.* Data complexity *of the compliance problem assumes that all* $\mathcal{O}$*,* $\mathcal{M}$ *and p are fixed, while only* $\mathcal{D}$ *forms the input.*

In our example, the data integration setting $(\mathcal{O}^{ex}, \mathcal{M}^{ex}, \mathcal{D}_1^{ex})$ does not comply to policy p^{ex} since $p^{ex}(P1)$ and $p^{ex}(P2)$ hold over all source instances indistinguishable from $\mathcal{D}_{ex}^1$ for $\mathcal{O}^{ex}$ and $\mathcal{M}^{ex}$ (and in particular they hold over $\mathcal{D}_2^{ex}$).

3.3. Strong Compliance

Definition 5 ensures that no *positive information* about the policy can be disclosed. It does not, however, preclude a malicious attacker from finding out potentially valuable *negative information* about the policy. For instance, even if compliance for our example policy is ensured, an attacker may still be able to find out that a specific hospital patient is *not* an oncology patient. This information can be valuable since, by ruling out many patients, the likelihood that the remaining ones are indeed oncology patients is increased (even if nothing certain can be said about them). In particular, consider the following source instance $\mathcal{D}_3^{ex}$.

$\mathcal{D}_3^{ex}$:

OncologyAppointment	*PatientID*	*DoctorID*	*Date*	*Treatment*
	*P*1	*D*1	30-11-2017	Daunorubicin

PaediatricsAppointment	*PatientID*	*DoctorID*	*Date*	*Location*
	*P*3	*D*3	15-10-2017	Room 225
	*P*4	*D*1	30-11-2017	Room 230

Instance $\mathcal{D}_3^{ex}$ induces the following virtual image over the global relational names.

$\mathcal{V}_{\mathcal{M},\mathcal{D}_3^{ex}}$:

Appointment	*PatientID*	*DoctorID*	*Date*
	*P*1	*D*1	30-11-2017
	*P*3	*D*3	15-10-2017
	*P*4	*D*1	30-11-2017

OncologistRecord	*DoctorID*	*Date*
	*D*1	30-11-2017

PaediatricianRecord	*DoctorID*	*Date*
	*D*2	15-10-2017
	*D*1	30-11-2017

We can observe that $(\mathcal{O}^{ex}, \mathcal{M}^{ex}, \mathcal{D}_3^{ex})$ complies to our example policy p^{ex}. This is because doctor $D1$ had both oncology and paediatrics appointments on the 30th of November and hence an attacker cannot determine with certainty that patient $P1$ (or $P4$) is an oncology patient; in particular, there is a source instance indistinguishable from $\mathcal{D}_3^{ex}$ in which the oncology appointments table has an entry for patient $P4$ visiting doctor $D1$ on the 30th of November and the paediatrics appointment table has an entry for patient $P1$ visiting the same doctor on the same day. Although this example satisfies the requirements of Definition 5, an attacker can obtain valuable information about the policy, namely that patient $P3$ did *not* have an oncology appointment; indeed, $P3$ had appointments only with a doctor $D3$ that we know did not have any oncology appointments. Consequently, an attacker may find out that $P1$ and $P4$ are the only possible oncology patients despite the fact of not being able to identify for sure which of them is in the oncology appointment relation.

In order to meet the requirements of applications with stricter privacy requirements, we introduce a stronger notion of compliance ensuring that neither positive nor negative information about the policy is disclosed.

Definition 6. *A data integration setting* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ strongly complies to *a Boolean policy* p *if there exists a source instance* $\mathcal{D}'$ *indistinguishable from* $\mathcal{D}$ *for* $\mathcal{O}$ *and* $\mathcal{M}$ *such that* $\mathcal{D} \models p$ *if and only if* $\mathcal{D}' \not\models p$. *Furthermore,* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ strongly complies *to a policy* $p(\mathbf{x})$ *with* n *free variables* $\mathbf{x}$ *if it strongly complies to* $p(\mathbf{a})$ *for all* n*-tuples* $\mathbf{a}$ *of constants from* $\mathcal{D}$.

The associated decision problem for an ontology language $\mathbb{O}$, *mapping language* $\mathbb{M}$, *and policy language* $\mathbb{P}$ *is defined as follows, where* $\mathcal{O}$ *ranges over* $\mathbb{O}$, $\mathcal{M}$ *over* $\mathbb{M}$, *and* p *over* $\mathbb{P}$: StrComply$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *is* `true` *if and only if* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ *strongly complies to* p. Data complexity *of the strong compliance problem assumes that all* $\mathcal{O}$, $\mathcal{M}$ *and* p *are fixed, while only* $\mathcal{D}$ *forms the input.*

We can check that our example data integration setting $(\mathcal{O}^{ex}, \mathcal{M}^{ex}, \mathcal{D}_3^{ex})$ does not strongly comply to our example policy p^{ex}. Indeed, $p^{ex}(P3)$ does not hold over all source instances indistinguishable from $\mathcal{D}_3^{ex}$ for $\mathcal{O}^{ex}$ and $\mathcal{M}^{ex}$.

3.4. Data-Independent Compliance

Finally, observe that both Definitions 5 and 6 provide guarantees for a specific source instance only. In particular, a compliant policy may become non-compliant if the data in the source instances is modified (even if neither the ontology nor the mappings change). For example, if we recall our example ontology $\mathcal{O}^{ex}$ mappings $\mathcal{M}^{ex}$ and policy p^{ex}, we already determined that compliance holds for source instance $\mathcal{D}_3^{ex}$ but not for $\mathcal{D}_1^{ex}$. Since database updates are relatively frequent operations, it makes sense in practice to consider also data-independent notions of compliance.

Definition 7. *Let* $\mathbb{O}$ *be an ontology language, let* $\mathbb{M}$ *be a mapping language, and let* $\mathbb{P}$ *be a policy language. We define the following decision problems, where* $\mathcal{O}$ *ranges over* $\mathbb{O}$, $\mathcal{M}$ *over* $\mathbb{M}$, *and* p *over* $\mathbb{P}$:

- ComplyAll$(\mathcal{O}, \mathcal{M}, p)$ *is* `true` *if and only if* Comply$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *is* `true` *for all source instances* $\mathcal{D}$; *and*
- StrComplyAll$(\mathcal{O}, \mathcal{M}, p)$ *is* `true` *if and only if* StrComply$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *is* `true` *for all source instances* $\mathcal{D}$.

In order to ensure that our example policy, mappings, and ontology satisfy the compliance requirements for every source instance, we could remove from $\mathcal{O}^{ex}$ the TGDs (5) and (6), and replace mappings (2) and (4) with the following mappings:

$$\mathsf{OncologyAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Treatment}) \rightarrow \mathsf{DoctorRecord}(\mathit{DoctorID}, \mathit{Date}), \tag{8}$$

$$\mathsf{PaediatricsAppointment}(\mathit{PatientID}, \mathit{DoctorID}, \mathit{Date}, \mathit{Location}) \rightarrow \mathsf{DoctorRecord}(\mathit{DoctorID}, \mathit{Date}). \tag{9}$$

With this modification, attackers can no longer query for the relations OncologistRecord and PaediatricianRecord and hence they are no longer able to exploit that information in order to discern whether an appointment in the global relation Appointment stems from an oncology or a paediatrics appointment in the source instance.

Note that there is no notion of data complexity for the problems in Definition 7 since the data is not part of the input. Furthermore, since the problems are data-independent, we assume that ontologies and mappings are constant-free and that policies are constant-free and Boolean.

3.5. Role of Ontology

The main purpose of an ontology in data integration is, on the one hand, to provide a common vocabulary for users to formulate queries against disparate sources and, on the other hand, to enrich query answers with information that is not explicitly stated in the source instances. In this section we argue that the presence of an ontology also impacts information disclosure.

We start by discussing the influence of ontology axioms on source indistinguishability. For this, let us consider once again our running example as well as the following source instance $\mathcal{D}_4^{ex}$:

$\mathcal{D}_4^{ex}$:

OncologyAppointment	*PatientID*	*DoctorID*	*Date*	*Treatment*
	*P*1	*D*1	30-11-2017	Daunorubicin
	*P*2	*D*2	10-06-2017	Docetaxel
	*P*3	*D*3	15-10-2017	Flutamide

PaediatricsAppointment	*PatientID*	*DoctorID*	*Date*	*Location*
	*P*3	*D*3	15-10-2017	Room 225
	*P*4	*D*4	10-08-2017	Room 230

Clearly, instances $\mathcal{D}_1^{ex}$ and $\mathcal{D}_4^{ex}$ are not indistinguishable for $\mathcal{O}^{ex}$ and $\mathcal{M}^{ex}$ since $\exists \mathit{Date}.\,\mathsf{OncologistRecord}(D3, \mathit{Date})$ holds over $(\mathcal{O}^{ex}, \mathcal{M}^{ex}, \mathcal{D}_4^{ex})$, but not over $(\mathcal{O}^{ex}, \mathcal{M}^{ex}, \mathcal{D}_1^{ex})$. Assume, however, that we now extend $\mathcal{O}^{ex}$ with the TGD

$$\mathsf{PaediatricianRecord}(\mathit{DoctorID}, \mathit{Date}) \rightarrow \mathsf{OncologistRecord}(\mathit{DoctorID}, \mathit{Date}) \tag{10}$$

stating that every paediatrician record is also an oncologist record (i.e., consider the situation where the database about paediatrics actually refers only to appointments made by specialists in oncological paediatrics). Then, it can be checked that $\mathcal{D}_1^{ex}$ and $\mathcal{D}_4^{ex}$ become indistinguishable. Our example shows that the addition of a TGD to an ontology can cause indistinguishability of two instances. The following proposition establishes that the converse is not possible—that is, if two source instances are indistinguishable, they will remain so regardless of any extension of the ontology.

Proposition 8. *Let $\mathcal{M}$ be a set of mappings, let $\mathcal{O}$ and $\mathcal{O}'$ be ontologies such that $\mathcal{O} \subseteq \mathcal{O}'$, and let $\mathcal{D}_1$ and $\mathcal{D}_2$ be source instances. If $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable for $\mathcal{O}$ and $\mathcal{M}$, then they are also indistinguishable for $\mathcal{O}'$ and $\mathcal{M}$.*

Proof. Assume that $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable for $\mathcal{O}$ and $\mathcal{M}$. By Lemma 3 we have that that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_1}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_2}$ are logically equivalent. By the monotonicity of first-order logic $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_1} \models \mathcal{V}_{\mathcal{M},\mathcal{D}_2}$ implies $\mathcal{O}' \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_1} \models \mathcal{V}_{\mathcal{M},\mathcal{D}_2}$ and, conversely, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_2} \models \mathcal{V}_{\mathcal{M},\mathcal{D}_1}$ implies $\mathcal{O}' \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_2} \models \mathcal{V}_{\mathcal{M},\mathcal{D}_1}$; as a result, $\mathcal{O}' \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_1}$ and $\mathcal{O}' \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_2}$ are also logically equivalent. Consequently, again by Lemma 3, $\mathcal{D}_1$ is also indistinguishable from $\mathcal{D}_2$ for $\mathcal{O}'$ and $\mathcal{M}$, as required. □

Our previous example also illustrates the influence of ontologies on compliance checking. In particular, recall that our example policy p^{ex} does not comply to the data integration setting $(\mathcal{O}^{ex}, \mathcal{M}^{ex}, \mathcal{D}_1^{ex})$. However, if we extend $\mathcal{O}^{ex}$ with (10) and

$$\mathsf{OncologistRecord}(\mathit{DoctorID}, \mathit{Date}) \rightarrow \mathsf{PaediatricianRecord}(\mathit{DoctorID}, \mathit{Date}), \tag{11}$$

then p^{ex} becomes compliant, since there is an instance indistinguishable from $\mathcal{D}_1^{ex}$ where the relation OncologyAppointment is empty; thus, it is no longer possible for an attacker to determine whether $p^{ex}(P1)$ or $p^{ex}(P2)$ hold in the source instance. Using Proposition 8 we can show that the converse situation cannot occur: if compliance (or strong compliance) holds, then it continues to hold for any extended ontology; this immediately implies that the same holds for data-independent versions of the problems.

Proposition 9. *If a data integration setting $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ complies (or strongly complies) to a policy p, then, for any ontology $\mathcal{O}'$ such that $\mathcal{O} \subseteq \mathcal{O}'$, the setting $(\mathcal{O}', \mathcal{M}, \mathcal{D})$ also complies (or strongly complies, respectively) to p.*

Proof. For the compliance case, assume that $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ complies to p. This means that for every tuple of constants $\mathbf{a}$ there must exist a source instance $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ for $\mathcal{O}$ and $\mathcal{M}$ such that $\mathcal{D}' \not\models p(\mathbf{a})$. By Proposition 8, $\mathcal{D}$ and $\mathcal{D}'$ are also indistinguishable for $\mathcal{O}'$ and $\mathcal{M}$ and hence $(\mathcal{O}', \mathcal{M}, \mathcal{D})$ complies to p, as required.

The proof for the strong compliance case is analogous, the only difference is that the indistinguishable $\mathcal{D}'$ is such that $\mathcal{D} \models p(\mathbf{a})$ if and only if $\mathcal{D}' \not\models p(\mathbf{a})$. □

4. Summary of Results

Our core technical results in Sections 5 and 6 provide a complete picture of the computational complexity of indistinguishability, compliance, and strong compliance for all combinations of ontology, mapping and policy languages introduced in Section 2. These results are summarised for the convenience of the reader in Table 1 and come with forward pointers to the relevant theorems and propositions. The results are clustered into ranges (column groups going across) that satisfy a given complexity bound. The most expressive combination in the range is denoted with "upper", and the least expressive with "lower". Note that there can be multiple column groups with the same bound (e.g., for Π_2^p combined complexity), since these represent cases incomparable in expressiveness. All the results in the table are given for the empty ontology, and the results relevant for standard OBDA mappings—that is, (possibly restricted) GAV mappings with global arity and frontier bounded by 2—are highlighted in bold font. For non-trivial ontology languages, the complexity of standard CQ (or fact) entailment may dominate over that of indistinguishability or compliance. In those cases, the results in the table should be amended as given next.

SOURCEIND

Combined complexity

complexity bound	upper	lower	upper	lower	upper	lower	upper	lower	upper	lower	upper	lower
mappings	GLAV	GLAV	GLAV	GLAV	GAV	CQ views	GAV	CQ views	LAV	LAV	GAV and LAV	CQ views and LAV
source arity	u.	b.	u.	b.	u.	b.	u.	b.	u.	b.	u.	b.
global arity	u.	b.	u.	b.	u.	u.	b.	b.	u.	b.	u.	b.
frontier	u.	u.	b.	b.	u.	u.	b.	b.	u.	b.	u.	b.
	Π_2^p		$\mathrm{P}^{\parallel\mathrm{NP}}$		Π_2^p		$\mathbf{P}^{\parallel\mathbf{NP}}$		NP		$\mathbf{AC}^0$	
	12*(2a)*	14*(a)*	12*(3a)*	15	12*(2a,b)*	14*(b)*	12*(3a,b)*	15	10*(2a)*	13*(a)*	10*(2b)*	

Data complexity

complexity bound	upper	lower	upper	lower	upper	lower	upper	lower
mappings	GLAV	GLAV	GAV	CQ views	LAV	LAV	GAV and LAV	CQ views and LAV
	AC^0		$\mathbf{AC}^0$		AC^0		$\mathbf{AC}^0$	
	10*(3a)*		10*(3a,b)*		10*(3a)*		10*(3a,b)*	

COMPLY **and** STRCOMPLY

Combined complexity

complexity bound		upper	lower	upper	lower	upper	lower	upper	lower	upper	lower	upper	lower	upper	lower
mappings		GLAV	GLAV	GLAV	GLAV	GAV	CQ views	GAV	CQ views	LAV	LAV	GAV and LAV	CQ views and LAV	GAV and LAV	CQ views and LAV
source arity		u.	b.	u.	b.	u.	b.	u.	b.	u.	b.	u.	u.	b.	b.
global arity		u.	b.	u.	b.	u.	u.	b.	b.	u.	b.	u.	b.	b.	b.
frontier		u.	u.	b.	b.	u.	u.	b.	b.	u.	b.	u.	b.	b.	b.
policy CQ	all	NEXPTIME		Π_3^p		NEXPTIME		$\mathbf{\Pi_3^p}$		Π_3^p		$\mathbf{\Pi_3^p}$		$\mathbf{\Pi_3^p}$	
		19*(1a)*	24*(a)*	19*(2a)*	25*(1)*	19*(1a,b)*	24*(b)*	19*(2a,b)*	25*(1)*	21*(1a)*	25*(1)*	21*(1a,b)*	25*(1)*	21*(1a,b)*	25*(1)*
	bounded arity	NEXPTIME		Σ_2^p		NEXPTIME		$\mathbf{\Sigma_2^p}$		Σ_2^p		$\mathbf{\Sigma_2^p}$		$\mathbf{\Sigma_2^p}$	
		19*(1a)*	24*(a)*	19*(3a)*	25(2), 26	19*(1a,b)*	24*(b)*	19*(3a,b)*	25*(2)*, 26	21*(2a)*	25(2)	21*(2a,b)*	25(2)	21*(2a,b)*	25(2)
	ground	NEXPTIME		Σ_2^p		NEXPTIME		$\mathbf{\Sigma_2^p}$		NP		**NP**		$\mathbf{AC}^0$	
		19*(1a)*	24*(a)*	19*(3a)*	26	19*(1a,b)*	24*(b)*	19*(3a,b)*	26	21*(3a)*	23*(a,b)*, 28	21*(3a,b)*	23*(b)*, 28	22*(b)*	

Data complexity

complexity bound		upper	lower	upper	lower	upper	lower	upper	lower
mappings		GLAV	GLAV	GAV	CQ views	LAV	LAV	GAV and LAV	CQ views and LAV
policy CQ	all	NP		**NP**		NP		**NP**	
		19*(4a)*	23*(a,b)*, 29*(a,b)*	19*(4a,b)*	29*(a,b)*	19*(4a)*	23*(a,b)*, 29*(b)*	19*(4a,b)*	29*(b)*
	ground	NP		**NP**		AC^0		$\mathbf{AC}^0$	
		19*(4a)*	23*(a,b)*, 29*(a)*	19*(4a,b)*	29*(a)*	22*(a)*		22*(a,b)*	

Table 1: Summary of complexity results for SOURCEIND, COMPLY, and STRCOMPLY, for the empty ontology. For combined complexity, "u." and "b." denote "unbounded" and "bounded", respectively, as the restrictions on source arity, global arity and frontier. All complexity bounds are tight, with the references to the corresponding (cases of) theorems and propositions below the bounds: for example, the first NEXPTIME-completeness of COMPLY for the most general setting (top left in the corresponding table) is by *Case (1a)* of Theorem 19 (upper bound) and *Case (a)* or *(b)* of Theorem 24 (two incomparable matching lower bounds).

– For the ontology language of full TGDs, all the combined complexity results below EXPTIME become EXPTIME by

  - *Cases (1), (2a)* and *(2b)* of Theorem 10 for the upper bound of SOURCEIND,
  - *Cases (a)* and *(b)* of Proposition 13 for the lower bound of SOURCEIND,
  - *Cases (2a)–(3b)* of Theorem 19, *Cases (1a)–(3b)* of Theorem 21, and *Cases (a)* and *(b)* of Theorem 22 for the upper bounds of COMPLY and STRCOMPLY, and
  - *Cases (a)* and *(b)* of Proposition 23 for the lower bounds of COMPLY and STRCOMPLY; and

  the AC^0 data complexity results become P by

  - *Cases (3a)* and *(3b)* of Theorem 10 for the upper bound of SOURCEIND,
  - *Cases (a)* and *(b)* of Proposition 13 for the lower bound of SOURCEIND,
  - *Cases (a)* and *(b)* of Theorem 22 for the upper bound of COMPLY and STRCOMPLY, and
  - *Case (b)* of Proposition 23 for the lower bound of COMPLY and STRCOMPLY.

– For the ontology language of linear TGDs, all the combined complexity results below PSPACE become PSPACE by the same theorems and propositions as for full TGDs, and the AC^0 data complexity results become P also by the same theorems and propositions as for full TGDs.

– For DL-Lite$_{\mathcal{R}}$ (and thus for OBDA), the AC^0 combined complexity becomes NLOGSPACE by

  - *Case (2b)* of Theorem 10 for the upper bound of SOURCEIND,
  - *Case (b)* of Proposition 13 for the lower bound of SOURCEIND,
  - *Case (1)* of Theorem 22 for the upper bound of COMPLY and STRCOMPLY, and
  - *Case (b)* of Proposition 23 for the lower bound of COMPLY and STRCOMPLY.

  The results without bounded global arity are not applicable in this setting and the data complexity results stay as for the empty ontology.

We next summarise the main take-away messages from our results in the table.

– Indistinguishability checking is decidable and we can obtain tractability in AC^0 in data complexity whenever the data complexity of reasoning in the ontology language of choice is also in AC^0. Indeed, the fundamental characterisation of source indistinguishability in terms of logical entailment provided by Lemma 3 in Section 3 suggests a basic algorithm, the analysis of which provides the relevant upper bounds. In the first step, the algorithm constructs the virtual images of the input mappings and source instances; then, in the second step, it checks whether the computed virtual images are logically equivalent with respect to the input ontology.

– Despite its simplicity, the complexity analysis of the aforementioned indistinguishability checking algorithm is rather subtle. For instance, it is interesting to observe that, for conventional OBDA where the arity of the global schema is bounded to 2, source indistinguishability can be checked in $P^{\|NP}$—the class of problems solvable in polynomial time using non-adaptive calls to an NP oracle; however, considering unbounded arity of global relational names immediately yields a complexity jump to the second level of the polynomial hierarchy.

– We obtain matching lower bounds for source indistinguishability checking for all relevant cases. Many of these bounds (in particular those for the empty ontology, where indistinguishability checking is often harder than query entailment for the ontology language of choice) are rather challenging to show and are interesting in their own right.

– Compliance checking is a more computationally demanding task than source indistinguishability. Although COMPLY and STRCOMPLY are also decidable for any ontology language with decidable query entailment problem, tractability in data complexity is much more limited and requires significant restrictions on both the mappings and the policy. To establish decidability of compliance checking, it suffices to observe that we only need to consider finitely many candidate source instances that are compatible with the information observable by an attacker. Essentially, we are only interested in the (finitely many) different ways in which the body of the mappings can match to the source instance to yield information in the observable virtual image. Intuitively, such candidate source instances can be guessed and the policy subsequently checked against each of them, although in many cases there is a need for guessing candidate source instances of exponential size.

– As in the case of indistinguishability checking, the detailed complexity analysis for all relevant cases of the generic compliance checking algorithm we just sketched in the previous point is rather subtle. Furthermore, if the input ontology consists of linear TGDs (as is the case of standard OBDA), then the generic algorithm does not provide optimal upper bounds and we need a specialsed algorithm.

– Matching lower bounds for compliance checking are provided for all cases. As for source indistinguishability, the lower bounds for the cases where query entailment is not the dominating source of complexity are rather challenging to prove and interesting in their own right. For instance, the NEXPTIME lower bound in Theorem 24, which is based on an involved reduction from a suitable variant of the domino tiling problem, shows that guessing exponential size candidate source instances when checking compliance is unavoidable even if we take the ontology to be empty and impose very strict requirements on the mappings. The lower bounds in Theorems 19 and 24 also contradict some of the upper bounds provided in [66] and show that the problems investigated there are actually much harder than previously thought. Our lower bounds in Theorems 25 and 26 also deserve especial attention since they transfer directly to the instance-based determinacy problem and close some existing open problems in the theory of data pricing [54] (see Section 8.2 for further details).

Section 7 presents our results on the data-independent problem COMPLYALL. Our first result is negative, showing that the problem is undecidable. While undecidability for a rich ontology and mapping language was expected, we show a more suprising result in Theorem 30: the problem is undecidable even for empty ontologies and GAV mappings of bounded source and global arities; furthermore, undecidability extends to CQ views as mappings if we allow for any of the ontology languages mentioned in this paper. These undecidability results depend on a rather sophisticated encoding of the halting problem for Turing machines, where non-compliant instances represent halting computations.

Then, we go on to show that, in the absence of an ontology and when mappings are restricted to CQ views, the problem becomes decidable. This is a surprising result, which makes a novel use of the idea of constructing a "critical instance" from prior work [4, 7, 28, 39, 63, 70]. In particular, we show that we can reduce the data-independent compliance problem to the data-dependent one on the aforementioned critical instance and that this reduction yields a (tight) CONP upper bound.

We leave open the decidability of COMPLYALL for LAV mappings. A related result has been proved in [6, Theorem 41], but where only source instances not satisfying p are considered, and the policy query is allowed to be a UCQ rather than a CQ.

More importantly, we leave the computational properties of the data-independent problem STRCOMPLYALL open. Our attempts to establish (un)decidability of this problem have so far been unsuccessful. We conjecture that the undecidability results of Theorem 30 and Corollary 38 extend to STRCOMPLYALL; our construction showing undecidability is, however, not sufficient and it is unclear whether it can be suitably modified. We thus believe that STRCOMPLYALL is an interesting and very challenging problem to be tackled in future work.

5. Source Indistinguishability

In this section we study the complexity of the source indistinguishability problem—that is, the problem of checking whether two given source instances are indistinguishable from the point of view of users of a data integration system. The results in this section will be relevant to the study of policy compliance later on. Source indistinguishability is also an interesting problem in its own right. On the one hand, it has obvious applications in practice, e.g., it can be used to determine whether given changes in the source instances can affect applications querying the system; on the other hand, as discussed in our related work section, indistinguishability problems akin to ours have recently received significant attention in the ontology literature motivated by applications in ontology reuse, ontology integration, and modular ontology design [11, 18, 51].

5.1. Decidability and Complexity Upper Bounds

We begin our study by providing upper bounds for the indistinguishability problem. In the next section we will show that all these bounds are tight.

Let us recall Lemma 3 in Section 3, which provides a fundamental characterisation of source indistinguishability in terms of logical entailment. The lemma suggests a basic algorithm that decides $\text{SOURCEIND}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ for any ontology language with decidable CQ entailment problem: in the first step, the algorithm constructs the virtual images $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$; then, in the second step, it checks whether $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ are equivalent.

Analysis of this algorithm reveals that source indistinguishability is no harder than CQ entailment in many cases; for instance, if the ontology language has sufficiently high complexity for CQ entailment then this complexity dominates over that of constructing the virtual images.

Theorem 10. *Problem* $\text{SOURCEIND}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *for* $\mathcal{O}$ *in an ontology language* $\mathbb{O}$ *and* $\mathcal{M}$ *in a mapping language* $\mathbb{M}$ *is in*

(1) EXPTIME*, if* $\mathbb{O}$ *has CQ entailment in* EXPTIME *with* P *data complexity; and*

(2) a complexity class C, for $C \in \{\text{AC}^0, \text{NLogSpace}, \text{P}, \text{NP}, \text{PSpace}, \text{ExpTime}\}$, if

- *(a) $\mathbb{O}$ has CQ entailment in C, and $\mathbb{M}$ is LAV, or*
- *(b) $\mathbb{O}$ has fact entailment in C, and $\mathbb{M}$ is GAV and LAV.*

In data complexity, $\text{SourceInd}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *for $\mathcal{O}$ in an ontology language $\mathbb{O}$ and $\mathcal{M}$ in a mapping language $\mathbb{M}$ is in*

(3) a complexity class C, for $C \in \{\text{AC}^0, \text{NLogSpace}, \text{P}, \text{NP}, \text{PSpace}, \text{ExpTime}\}$, if

- *(a) $\mathbb{O}$ has CQ entailment in C in data complexity, or*
- *(b) $\mathbb{O}$ has fact entailment in C in data complexity, and $\mathbb{M}$ is GAV.*

Before proving the theorem, we note that the set of complexity classes that C is ranging over in *Cases (2)* and *(3)* is motivated just by the CQ and fact entailment problems for the ontology languages considered in this paper. This set can be extended by many other standard complexity classes, such as NExpTime—the only requirement is that the class should be closed under AC^0 reductions and intersection (the proofs of the statements would be the same). *Case (1)* can be relativised in the same way: ExpTime can be replaced with any usual complexity class subsuming ExpTime, such as NExpTime and ExpSpace; in this case, however, we need stronger, more difficult to formulate, closure properties, which we omit here. Finally, recall from Section 2 that the requirement in *Case (1)* means the existence of an algorithm whose running time is bounded exponentially in general and polynomially in the size of data; for example, *Case (1)* gives us an ExpTime upper bound for indistinguishability when $\mathbb{O}$ is the language of full TGDs.

Proof. *Case (1)* follows by analysis of our basic algorithm based on Lemma 3, which first constructs $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$, and then checks equivalence of $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$. Let s_o, s_m, s_d and $s_{d'}$ be the sizes of (binary representations of) $\mathcal{O}$, $\mathcal{M}$, $\mathcal{D}$ and $\mathcal{D}'$, respectively. There exist polynomials $\mathtt{p}_1$ and $\mathtt{p}_2$ such that (the representations of) $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ can be constructed in time $s_v = \mathtt{p}_1(s_m + s_d + s_{d'})^{\mathtt{p}_2(s_m)}$, and, therefore, the sizes of $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ are bounded by s_v as well. Then, equivalence of $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ can be checked by means of s_v CQ entailments, each of which can be done, for polynomials $\mathtt{p}_3$ and $\mathtt{p}_4$, in time $\mathtt{p}_3(s_o + s_m + s_v)^{\mathtt{p}_4(s_o+s_m)}$ (since CQ entailment is in ExpTime and in P in data complexity for $\mathbb{O}$). So, overall, indistinguishability can be checked in time $s_v + s_v \cdot \mathtt{p}_3(s_o + s_m + s_v)^{\mathtt{p}_4(s_o+s_m)}$, as required.

Case (2a) follows by analysis of the same algorithm. Since entailment of each CQ in $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ by $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ can be checked in C (and the same holds for $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$) and C is closed under AC^0 reductions and intersection, checking equivalence of $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ can also be done in C. Since the mappings $\mathcal{M}$ are LAV, $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ are of polynomial size in the sizes of $\mathcal{M}$, $\mathcal{D}$ and $\mathcal{D}'$, and it is possible to construct them in AC^0. Therefore, since C is closed under AC^0 reductions, the overall algorithm for indistinguishability in is C as well.

Cases (2b), (3a), and *(3b)* can be proved in the same way as *Case (2a)*. The only difference is that in *Cases (2b)* and *(3b)* the mappings $\mathcal{M}$ are GAV, so we are interested in fact entailment instead of CQ entailment, while in *Cases (3a)* and *(3b)* we are interested in data complexity, so linearity of $\mathcal{M}$ is not necessary. □

If the ontology consists of linear TGDs, as is the case in standard OBDA settings, the basic algorithm is in fact not optimal. In particular, we can avoid the explicit construction of the virtual images by exploiting the following property of linear ontologies: to check whether $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models q$ with a Boolean CQ q in $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ it suffices to consider only a set of instantiations of the frontier of $\mathcal{M}$ over $\mathcal{D}$ with the size polynomially bounded in the size of q.

Lemma 11. *Let $\mathcal{O}$ be a linear ontology, $\mathcal{M}$ be a set of mappings, and $\mathcal{D}$, $\mathcal{D}'$ be source instances. Then, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ if and only if the following condition holds: for each mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and tuple of constants $\mathbf{a}, \mathbf{c}$ with (all facts of) $\varphi(\mathbf{a}, \mathbf{c})$ in $\mathcal{D}'$ there are an integer k bounded by the number $|\psi|$ of atoms in ψ, mappings $\varphi_i(\mathbf{x}_i, \mathbf{z}_i) \rightarrow \exists \mathbf{y}_i.\, \psi_i(\mathbf{x}_i, \mathbf{y}_i)$ in $\mathcal{M}$ and tuples of constants $\mathbf{a}_i, \mathbf{c}_i$, for $i = 1, \ldots, k$, such that $\varphi_i(\mathbf{a}_i, \mathbf{c}_i)$ is in $\mathcal{D}$ for all i and $\mathcal{O} \cup \{\exists \mathbf{y}_i.\, \psi_i(\mathbf{a}_i, \mathbf{y}_i) \mid i = 1, \ldots, k\} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$.*

Proof. We start with the backwards direction. For this, assume that the condition in the lemma holds. Pick a CQ $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ in $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$. By the definition of $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$, there exists a mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ such that $\mathcal{D}' \models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$. But then, there must exist a tuple of constants $\mathbf{c}$ such that all facts of $\varphi(\mathbf{a}, \mathbf{c})$ are in $\mathcal{D}'$. By the condition of the lemma, there exist mappings $\varphi_i(\mathbf{x}_i, \mathbf{z}_i) \rightarrow \exists \mathbf{y}_i.\, \psi_i(\mathbf{x}_i, \mathbf{y}_i)$ and tuples of constants $\mathbf{a}_i, \mathbf{c}_i$, for $i = 1, \ldots, k$, $k \leq |\psi|$, such that all facts of $\varphi_i(\mathbf{a}_i, \mathbf{c}_i)$ are in $\mathcal{D}$ for all i and $\mathcal{O} \cup \{\exists \mathbf{y}_i.\, \psi_i(\mathbf{a}_i, \mathbf{y}_i) \mid i = 1, \ldots, k\} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$. Since $\{\exists \mathbf{y}_i.\, \psi_i(\mathbf{a}_i, \mathbf{y}_i) \mid i = 1, \ldots, k\} \subseteq \mathcal{V}_{\mathcal{M},\mathcal{D}}$ we have that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$.

For the forward direction, assume that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \mathcal{V}_{\mathcal{M},\mathcal{D}'}$. Let $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ be a mapping in $\mathcal{M}$ and $\mathbf{a}, \mathbf{c}$ be a tuple of constants such that $\varphi(\mathbf{a}, \mathbf{c}) \subseteq \mathcal{D}'$. By the definition of $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ we have that $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y}) \in \mathcal{V}_{\mathcal{M},\mathcal{D}'}$, and hence by our assumption we have that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$. In the following argument, we use the properties of the classic chase procedure [47] applied to linear TGDs. In particular, we use the property that, for linear TGDs, each inference step in the chase procedure depends of at most one fact [47], and hence all facts in the chase are "reachable" from a single input fact. Now, the completeness of the chase ensures that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ implies the existence of a tuple of constants $\mathbf{b}$ such that $\psi(\mathbf{a}, \mathbf{b})$ is contained in

the chase of $\mathcal{O}$ union the input facts obtained from $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ by one-to-one substituting of all existentially quantified variables with fresh constants. Since $\mathcal{O}$ consists of linear TGDs, each fact in $\psi(\mathbf{a}, \mathbf{b})$ ultimately depends on a single input fact. By the definition of $\mathcal{V}_{\mathcal{M},\mathcal{D}}$, each input fact of the chase is justified by a mapping $\varphi_i(\mathbf{x}_i, \mathbf{z}_i) \rightarrow \exists \mathbf{y}_i.\, \psi_i(\mathbf{x}_i, \mathbf{y}_i)$ in $\mathcal{M}$ and tuples of constants $\mathbf{a}_i, \mathbf{c}_i$ such that all the facts of $\varphi_i(\mathbf{a}_i, \mathbf{c}_i)$ are in $\mathcal{D}$, which implies the claim. □

Lemmas 3 and 11 suggest a simple algorithm for checking SOURCEIND$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ if the input ontology $\mathcal{O}$ is linear:

1. for each mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and tuple $\mathbf{a}$ such that $\mathcal{D}' \models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$
   - check all sets of mappings $\varphi_i(\mathbf{x}_i, \mathbf{z}_i) \rightarrow \exists \mathbf{y}_i.\, \psi_i(\mathbf{x}_i, \mathbf{y}_i)$ in $\mathcal{M}$ and tuples of constants $\mathbf{a}_i$ with $i = 1, \ldots, k$, $k \leq |\psi|$, such that $\mathcal{D} \models \exists \mathbf{z}_i.\, \varphi_i(\mathbf{a}_i, \mathbf{z}_i)$ and return `false` if none of them satisfies $\mathcal{O} \cup \{\exists \mathbf{y}_i.\, \psi_i(\mathbf{a}_i, \mathbf{y}_i) \mid i = 1, \ldots, k\} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$;
2. repeat the previous step with $\mathcal{D}$ and $\mathcal{D}'$ swapped and return `true` afterwards.

By analysing this algorithm, we can provide upper bounds to source indistinguishability for several other relevant cases.

Theorem 12. *Problem* SOURCEIND$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *for $\mathcal{O}$ in an ontology language $\mathbb{O}$ and $\mathcal{M}$ in a mapping language $\mathbb{M}$ is in*

(1) PSPACE*, if $\mathbb{O}$ is linear;*

(2) Π_2^p*, if*

- *(a) $\mathbb{O}$ is linear and has CQ entailment in* NP*, or*
- *(b) $\mathbb{O}$ is linear and has fact entailment in* NP*, and $\mathbb{M}$ is GAV; and*

(3) $\mathrm{P}^{\|\mathrm{NP}}$*, if*

- *(a) $\mathbb{O}$ is linear and has CQ entailment in* NP*, and $\mathbb{M}$ is of bounded frontier, or*
- *(b) $\mathbb{O}$ is linear and has fact entailment in* NP*, and $\mathbb{M}$ is GAV and of bounded frontier.*

Before proving this theorem, consider the following remarks. First, according to *Case (3)* (either *(a)* or *(b)*), the indistinguishability problem is in $\mathrm{P}^{\|\mathrm{NP}}$ for standard OBDA settings, where mappings are GAV and the ontology is in DL-Lite$_{\mathcal{R}}$ (it is in AC^0 in data complexity by *Case (3b)* of Theorem 10). Second, *Cases (2)* and *(3)* can be relativized in a similar way as *Cases (2)* and *(3)* of Theorem 10; for example, if CQ entailment is in a complexity class C for $\mathbb{O}$ and NP $\subseteq C$, then SOURCEIND$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ is in the class coNP^C.

Proof. Case (1) follows immediately from the analysis of the algorithm: all mappings $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and tuples $\mathbf{a}$ are checked, one by one, whether either $\mathcal{D}' \not\models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$ or there is a set of mappings $\varphi_i(\mathbf{x}_i, \mathbf{z}_i) \rightarrow \exists \mathbf{y}_i.\, \psi_i(\mathbf{x}_i, \mathbf{y}_i)$ in $\mathcal{M}$ and tuples of constants $\mathbf{a}_i$ with $i = 1, \ldots, k$, $k \leq |\psi|$, such that $\mathcal{O} \cup \{\exists \mathbf{y}_i.\, \psi_i(\mathbf{a}_i, \mathbf{y}_i) \mid i = 1, \ldots, k\} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$; then the same is done for $\mathcal{D}$ and $\mathcal{D}'$ swapped. If CQ entailment is in PSPACE, then all this can be done in PSPACE as well.

Case (2a) is similar to *Case (1)*: to falsify indistinguishability, a mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and tuple $\mathbf{a}$ is guessed and then NP oracles are called two times, one to check whether $\mathcal{D}' \not\models \exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$ and the other to check whether there exists a set of mappings and a set of tuples satisfying the required properties, and then two more times for $\mathcal{D}$ and $\mathcal{D}'$ swapped. *Case (2b)* is the same as *Case (2a)*, except that $\mathbf{y}$ is empty, and hence we need to check for fact entailment over ontology instead of CQ entailment.

Cases (3a) and *(3b)* is also very similar: since the frontier—that is, the size of $\mathbf{a}$—is bounded, we do not need to guess $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and $\mathbf{a}$ because we can check all of them one by one in polynomial time. □

5.2. Complexity Lower Bounds

We begin our study of lower bounds by showing that source indistinguishability is at least as hard as CQ entailment under mild assumptions on the mapping language.

Proposition 13. *Problem* SOURCEIND$(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *is C-hard, for a complexity class C, when $\mathcal{O}$ ranges over an ontology language $\mathbb{O}$ and $\mathcal{M}$ over a mapping language $\mathbb{M}$ such that*

(a) CQ entailment is C-hard for $\mathbb{O}$, and $\mathbb{M}$ contains all sets of LAV mappings, or

(b) fact entailment is C-hard for $\mathbb{O}$, and $\mathbb{M}$ contains all sets of mappings that are CQ views and LAV;

if, additionally, the arity of the schema of any ontology in $\mathbb{O}$ is bounded by a number k, then, in both cases, mapping language $\mathbb{M}$ may be additionally restricted to have the frontier, source arity and global arity all bounded by k. The same holds if the problems are considered in data complexity.

In particular, recall that CQ and fact entailment are ExpTime- and PSpace-complete for full and linear ontologies, respectively, so these bounds propagate to the hardness of indistinguishability for these ontology languages. Moreover, this proposition provides matching lower bounds for all the cases of Theorem 10 and *Case (1)* of Theorem 12.

Proof. For *Case (a)*, consider an input to the CQ entailment problem consisting of an ontology $\mathcal{O}_0$, an instance $\mathcal{D}_0$, and a Boolean CQ q_0. Let $\mathcal{D}'_0$ be the source instance obtained from $\mathcal{D}_0$ by a one-to-one renaming of each relational name R with a fresh source relational name R' of the same arity. Furthermore, let Q be a fresh nullary source relational name. We define an input to SourceInd with an ontology $\mathcal{O} = \mathcal{O}_0$, source instances $\mathcal{D} = \mathcal{D}'_0 \cup \{\mathsf{Q}()\}$ and $\mathcal{D}' = \mathcal{D}'_0$, and mappings $\mathcal{M}$ consisting of a GAV and LAV mapping $\mathsf{R}'(\mathbf{x}) \rightarrow \mathsf{R}(\mathbf{x})$, for each relational name R in $\mathcal{D}_0$, and a LAV mapping $\mathsf{Q}() \rightarrow q_0$. We then argue that $\textsc{SourceInd}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}') = \texttt{true}$ if and only if $\mathcal{O}_0 \cup \mathcal{D}_0 \models q_0$. Assume that $\textsc{SourceInd}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}') = \texttt{true}$. Then, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ must be equivalent. By construction of $\mathcal{D}$ and $\mathcal{M}$, the set of Boolean CQs $\mathcal{V}_{\mathcal{M},\mathcal{D}} = \mathcal{V}_{\mathcal{M},\mathcal{D}'} \cup \{q_0\}$ and the equivalence imply that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'} \models q_0$. Since $\mathcal{O} = \mathcal{O}_0$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'} = \mathcal{D}_0$ we have that $\mathcal{O}_0 \cup \mathcal{D}_0 \models q_0$, as required. For the converse, assume $\mathcal{O}_0 \cup \mathcal{D}_0 \models q_0$. Since $\mathcal{O} = \mathcal{O}_0$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}'} = \mathcal{D}_0$, we have that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'} \models q_0$. Since $\mathcal{V}_{\mathcal{M},\mathcal{D}} = \mathcal{V}_{\mathcal{M},\mathcal{D}'} \cup \{q_0\}$ we have $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'} \models \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and hence $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ must be equivalent, which implies $\textsc{SourceInd}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}') = \texttt{true}$.

Case (b) can be proved in a similar way: the only differences are that $\mathcal{M}$ does not contain the LAV mapping, and that, instead of $\mathsf{Q}()$, source instance $\mathcal{D}$ contains the fact obtained from the fact in the input of the fact entailment problem by renaming the relational name in the same way as for the instances.

The same proofs work for the statements about bounded arity and data complexity. □

Checking indistinguishability can be, however, strictly harder than CQ entailment. CQ entailment in the absence of an ontology is in NP, while the following theorem proves that even with no ontology SourceInd can be Π^p_2-hard.

Theorem 14. *Problem* $\textsc{SourceInd}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *is* Π^p_2*-hard if* $\mathcal{O}$ *is empty and* $\mathcal{M}$ *ranges over an ontology language* $\mathbb{M}$ *such that either*

(a) $\mathbb{M}$ *contains all sets of mappings with source and global arity both bounded by 2, or*

(b) $\mathbb{M}$ *contains all sets of CQ views with source arity bounded by 2.*

This theorem provides matching lower bounds to *Case (2)* of Theorem 12. Note that there are two incomparable lower bounds, and, according to *Case (3)* of Theorem 12, if we join the conditions for these bounds then the problem becomes easier, in $\mathrm{P}^{\|\mathrm{NP}}$ (the matching lower bound for that case is shown in Theorem 15).

Proof. We start with *Case (b)* and show Π^p_2-hardness by reduction of the $\forall\exists$3Sat problem, and then explain how to adapt the reduction to *Case (a)*.

Let $\Phi = \forall\mathbf{u}.\, \exists\mathbf{v}.\, \Psi$, where Ψ is a propositional formula in 3CNF over variables $\mathbf{u} \cup \mathbf{v}$—that is, a conjunction of clauses of the form $\ell_1 \vee \ell_2 \vee \ell_3$ with propositional literals ℓ_1, ℓ_2, ℓ_3 over $\mathbf{u} \cup \mathbf{v}$ (i.e., each ℓ_j is either a variable in $\mathbf{u} \cup \mathbf{v}$ or the negation of such a variable). Let also $\mathbf{u} = u_1, \ldots, u_n$.

On the base of Φ we construct source instances $\mathcal{D}$ and $\mathcal{D}'$ as well as a set of CQ views $\mathcal{M}$ with source arity bounded by 2 such that Φ is $\texttt{true}$ if and only if $\textsc{SourceInd}(\emptyset, \mathcal{M}, \mathcal{D}, \mathcal{D}') = \texttt{true}$.

We first define the source schema of the setting. Let Cl_γ be a unary source relational name for each clause γ in Ψ, and let Arg_j for $j = 1, 2, 3$, and UVal be binary source relational names.

Next we define source instance $\mathcal{D}$, and start with describing the constants that $\mathcal{D}$ uses. Each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$ in Ψ mentions at most 3 variables, so there are at most 8 possible assignments of these variables with at most 7 of them satisfying γ. Instance $\mathcal{D}$ uses a constant a^π_γ for each such satisfying assignment π of variables of γ. Also, $\mathcal{D}$ uses constants f_w and t_w for each variable $w \in \mathbf{u} \cup \mathbf{v}$, representing assignments of w to $\texttt{false}$ and $\texttt{true}$, respectively. Finally, $\mathcal{D}$ uses two constants 0 and 1. Let $\mathcal{D}$ consist of the following facts over the introduced constants:

- $\mathsf{Cl}_\gamma(a^\pi_\gamma), \mathsf{Arg}_1(a^\pi_\gamma, b_1), \mathsf{Arg}_2(a^\pi_\gamma, b_2), \mathsf{Arg}_3(a^\pi_\gamma, b_3)$ for each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$ in Ψ and each satisfying assignment π of γ, where $b_j = f_w$ if $\pi(w) = \texttt{false}$ and $b_j = t_w$ if $\pi(w) = \texttt{true}$ for each $j = 1, 2, 3$ and w the variable in ℓ_j;
- $\mathsf{UVal}(f_u, 0)$ and $\mathsf{UVal}(t_u, 1)$ for all variables u in $\mathbf{u}$.

Source instance $\mathcal{D}'$ mentions the same constants as $\mathcal{D}$, except that it uses a constant a^π_γ for each assignment π of variables of each clause γ in Ψ (as opposed to each satisfying assignment in case of $\mathcal{D}$), so each γ has at most 8 such constants in $\mathcal{D}'$ (as opposed to up to 7 in case of $\mathcal{D}$). Hence, $\mathcal{D}'$ consists of the same facts as $\mathcal{D}$ except that it has facts $\mathsf{Cl}_\gamma(a^\pi_\gamma)$, $\mathsf{Arg}_1(a^\pi_\gamma, b_1)$, $\mathsf{Arg}_2(a^\pi_\gamma, b_2)$, $\mathsf{Arg}_3(a^\pi_\gamma, b_3)$ for each γ in Ψ and each assignment π of variables in γ (as opposed to each satisfying assignment).

The global schema consists of a single n-ary relational name Target.

Finally, we define $\mathcal{M}$ as consisting of a single GAV mapping, for w_γ^j the propositional variable in a literal ℓ_j with number $j = 1, 2, 3$ of a clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$,

$$\bigwedge_{\gamma \text{ in } \Psi} \left(\mathsf{Cl}_\gamma(z_\gamma) \wedge \bigwedge_{j=1}^{3} \mathsf{Arg}_j(z_\gamma, z_{w_\gamma^j}) \right) \quad \wedge \quad \bigwedge_{i=1}^{n} \mathsf{UVal}(z_{u_i}, x_{u_i}) \quad \rightarrow \quad \mathsf{Target}(x_{u_1}, \ldots, x_{u_n})$$

over variables z_γ for γ in Ψ, z_w for $w \in \mathbf{u} \cup \mathbf{v}$, and x_u for $u \in \mathbf{u}$.

We argue that Φ is `true` if and only if $\textsc{SourceInd}(\emptyset, \mathcal{M}, \mathcal{D}, \mathcal{D}') = \texttt{true}$. According to the mapping, $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ contains facts $\mathsf{Target}(c_1, \ldots, c_n)$ for all $c_i \in \{0, 1\}$ and $1 \leq i \leq n$. It is routine to check that $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ consists of exactly the same facts (thus making $\mathcal{D}$ and $\mathcal{D}'$ indistinguishable) if and only if Φ is `true`. Indeed, by construction, $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ contains $\mathsf{Target}(c_1, \ldots, c_n)$ with each $c_i \in \{0, 1\}$ if and only if there exists an assignment of all variables $w \in \mathbf{u} \cup \mathbf{v}$ that agrees with $c_1, \ldots, c_n$ on $\mathbf{u}$ and satisfies all the clauses in Ψ. Therefore, $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ contains all $\mathsf{Target}(c_1, \ldots, c_n)$ if and only if each assignment of $\mathbf{u}$ has an assignment of $\mathbf{v}$ turning Ψ to `true`.

The proof for *Case (b)* can be easily modified to work for *Case (a)*: the only difference is that instead of a single n-ary relational name Target the global schema consists of n binary relational names Target_i, $i = 1, \ldots, n$, and the head of the single mapping in $\mathcal{M}$ is not $\mathsf{Target}(x_{u_1}, \ldots, x_{u_n})$, but

$$\exists y.\, \mathsf{Target}_1(y, x_{u_1}) \wedge \cdots \wedge \mathsf{Target}_n(y, x_{u_n}).$$

Correctness of this reduction follows immediately from the correctness of the reduction for *Case (a)*. □

Finally, if we join the conditions of *Cases (a)* and *(b)* of Theorem 14 (as in DL-Lite$_\mathcal{R}$), the problem stays hard for $\mathrm{P}^{\|\mathrm{NP}}$.

Theorem 15. *Problem* $\textsc{SourceInd}(\mathcal{O}, \mathcal{M}, \mathcal{D}, \mathcal{D}')$ *is* $\mathrm{P}^{\|\mathrm{NP}}$*-hard if* $\mathcal{O}$ *is empty and* $\mathcal{M}$ *ranges over an ontology language that contains all sets of CQ views with source arity bounded by 2 and global arity bounded by 0.*

Recall that in this setting the bound on the global arity implies the same bound on the frontier. Note that this theorem provides the matching lower bound to *Case (3)* of Theorem 12, and hence is applicable to DL-Lite$_\mathcal{R}$ ontologies.

Proof. We show $\mathrm{P}^{\|\mathrm{NP}}$-hardness by reduction of MAX-TRUE-3SAT-EQUALITY, a known $\mathrm{P}^{\|\mathrm{NP}}$-complete problem [72]. The input of this problem consists of two satisfiable propositional formulae Ψ_1 and Ψ_2 in 3CNF and the question is whether $\max_{\texttt{true}}(\Psi_1) = \max_{\texttt{true}}(\Psi_2)$; here, for a satisfiable propositional formula Ψ in 3CNF, $\max_{\texttt{true}}(\Psi)$ is the maximum of the number of variables assigned to `true` over all satisfying assignments of Ψ.

For the reduction, we proceed as follows. First, for each propositional formula Ψ in 3CNF we construct a source instance $\mathcal{D}_\Psi$ and a set of CQ views $\mathcal{M}_\Psi$ with the required bounds on the frontier, source arity, and target arity. Second, we show, for two formulae Ψ_1 and Ψ_2, that $\textsc{SourceInd}(\emptyset, \mathcal{M}_{\Psi_1} \cup \mathcal{M}_{\Psi_2}, \mathcal{D}_{\Psi_1}, \mathcal{D}_{\Psi_2})$ is `true` if and only if $\max_{\texttt{true}}(\Psi_1) = \max_{\texttt{true}}(\Psi_2)$. At this point, even if $\mathcal{M}_\Psi$ is a set of CQ views, $\mathcal{M}_{\Psi_1} \cup \mathcal{M}_{\Psi_2}$ is not, because the global schema of $\mathcal{M}_\Psi$ does not depend on Ψ; therefore, on the final step we show how to modify the reduction in a way that $\mathcal{M}_{\Psi_1}$ and $\mathcal{M}_{\Psi_2}$ are the same set of mappings, which suffices for the proof.

Consider a propositional formula Ψ in 3CNF over variables $u_1, \ldots, u_n$, which is a conjunction of clauses of the form $\ell_1 \vee \ell_2 \vee \ell_3$ for ℓ_j either a propositional variable from $u_1, \ldots, u_n$ or the negation of such a variable; for each such clause γ and each $j = 1, 2, 3$ denote u_γ^j the variable in ℓ_j. In this proof, we assume that variables $u_1, \ldots, u_n$ are ordered; this order plays a technical role and can be arbitrary (but fixed).

Similarly to the Π_2^p-hardness proof of Theorem 14, the source schema has unary relational name Cl_γ^Ψ for each clause γ in Ψ as well as binary relational names Arg_j for $j = 1, 2, 3$, while source instance $\mathcal{D}_\Psi$ uses a constant a_γ^π for each satisfying assignment π of variables of each clause γ in Ψ and constants f_{u_i} and t_{u_i} for each variable u_i. Besides this, the source schema has binary relational names $\mathsf{PartSum}$, PreVal and $\mathsf{CurrVal}$, as well as unary relational names Sum_m, for $m = 1, \ldots, n$. Also, the source instance $\mathcal{D}_\Psi$ uses constants d_i^k for each $i = 1, \ldots, n$ and $k = 0, \ldots, i$, as well as constants c_i^{k0} and c_i^{k1} for each $i = 2, \ldots, n$ and $k = 0, \ldots, i-1$.

Instance $\mathcal{D}_\Psi$ consists of

- the facts $\mathsf{Cl}_\gamma^\Psi(a_\gamma^\pi), \mathsf{Arg}_1(a_\gamma^\pi, b_1), \mathsf{Arg}_2(a_\gamma^\pi, b_2), \mathsf{Arg}_3(a_\gamma^\pi, b_3)$ for each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$ in Ψ and each satisfying assignment π of γ, where, for each $j = 1, 2, 3$, $b_j = f_{u_\gamma^j}$ if $\pi(u_\gamma^j) = \texttt{false}$ and $b_j = t_{u_\gamma^j}$ if $\pi(u_\gamma^j) = \texttt{true}$;
- the facts $\mathsf{PartSum}(f_{u_1}, d_1^0)$ and $\mathsf{PartSum}(t_{u_1}, d_1^1)$;
- for each i and k such that $2 \leq i \leq n$ and $0 \leq k \leq i-1$, the facts

$$\mathsf{PreVal}(d_{i-1}^k, c_i^{k0}), \mathsf{CurrVal}(f_{u_i}, c_i^{k0}), \mathsf{PartSum}(c_i^{k0}, d_i^k),$$
$$\mathsf{PreVal}(d_{i-1}^k, c_i^{k1}), \mathsf{CurrVal}(t_{u_i}, c_i^{k1}), \mathsf{PartSum}(c_i^{k1}, d_i^{k+1}),$$

– the fact $\mathsf{Sum}_m(d_n^k)$, for each k and m such that $0 \le m \le k \le n$.

Let the global schema consists of nullary relational names Sum'_m, for $m = 1, \ldots, n$.

Finally, let $\mathcal{M}_\Psi$ consist of the following mappings, for each $0 \le m \le n$, over variables z_γ for γ in Ψ, z_{u_i} and s_i for $i = 1, \ldots, n$, and r_i for $i = 2, \ldots, n$:

$$\bigwedge_{\gamma \text{ in } \Psi} \left(\mathsf{Cl}_\gamma^\Psi(z_\gamma) \wedge \bigwedge_{j=1}^{3} \mathsf{Arg}_j(z_\gamma, z_{u_\gamma^j}) \right) \wedge$$

$$\mathsf{PartSum}(z_{u_1}, s_1) \wedge \bigwedge_{i=2}^{n} \Big(\mathsf{PreVal}(s_{i-1}, r_i) \wedge \mathsf{CurrVal}(z_{u_i}, r_i) \wedge \mathsf{PartSum}(r_i, s_i) \Big) \wedge \mathsf{Sum}_m(s_n) \rightarrow \mathsf{Sum}'_m().$$

The key property of $\mathcal{M}_\Psi$ and $\mathcal{D}_\Psi$ is that $\mathcal{V}_{\mathcal{M}_\Psi, \mathcal{D}_\Psi} \models \mathsf{Sum}'_m()$ for an integer m if and only if $m \le \max_{\mathtt{true}}(\Psi)$. Indeed, consider a satisfying assignment σ of Ψ. Note that the bodies of all the mappings in $\mathcal{M}_\Psi$ differ only in the last atoms $\mathsf{Sum}_m(s_n)$. As before, the Cl_γ^Ψ and Arg_j atoms of the body of any mapping can be uniquely mapped to $\mathcal{D}_\Psi$ in such that way that each z_{u_i} is mapped to f_{u_i} or t_{u_i} depending of the Boolean value $\sigma(u_i)$. If z_{u_1} is mapped to f_{u_1}, then variable s_1 can be sent only to d_1^0: intuitively, $\mathsf{PartSum}(f_{u_i}, d_1^0)$ in the image of the homomorphism represents that 0 variables are assigned to `true` among the first 1 variables. If z_{u_1} is mapped to t_{u_1} then s_1 is sent to d_1^1, representing a similar fact. Having a homomorphism defined for all the variables up to s_{i-1}, which is sent to d_{i-1}^k, we know that exactly k variables are assigned to `true` among the first $i-1$ ones. If the current variable u_i is assigned to `false` by σ—that is, z_{u_i} is sent to f_{u_i}—then r_i can be sent only to c_i^{k0}, and, hence, s_i is sent to d_i^k. Similarly, if u_i is assigned to `true`, then s_i is sent to d_i^{k+1}. At the end, s_n is sent to d_n^k where k is the total number of variables assigned to `true` by σ; since $\mathcal{D}_\Psi$ contains $\mathsf{Sum}_0(d_n^k), \ldots, \mathsf{Sum}_k(d_n^k)$, virtual image $\mathcal{V}_{\mathcal{M}_\Psi, \mathcal{D}_\Psi}$ contains atoms $\mathsf{Sum}'_0(), \ldots, \mathsf{Sum}'_k()$. This process goes through for all satisfying assignments, so the key property indeed holds.

Consequently, MAX-TRUE-3SAT-EQUALITY(Ψ_1, Ψ_2) is `true` if and only if SOURCEIND$(\emptyset, \mathcal{M}_{\Psi_1} \cup \mathcal{M}_{\Psi_2}, \mathcal{D}_{\Psi_1}, \mathcal{D}_{\Psi_2})$ is `true`.

As already mentioned, $\mathcal{M}_{\Psi_1}$ and $\mathcal{M}_{\Psi_2}$ have mappings with same heads, so even if $\mathcal{M}_{\Psi_1}$ and $\mathcal{M}_{\Psi_2}$ are sets of CQ views, their union is not. The difference between $\mathcal{M}_{\Psi_1}$ and $\mathcal{M}_{\Psi_2}$ is that the body of each of their mappings has the conjunction $\mathsf{Cl}_\gamma^{\Psi_k}(z_\gamma) \wedge \bigwedge_{j=1}^{3} \mathsf{Arg}_j(z_\gamma, z_{u_\gamma^j})$ for each clause, for $k = 1, 2$, while Ψ_1 and Ψ_2 may have different triples of variables in clauses. One of the ways to overcome this is to modify Ψ_1 and Ψ_2 such that the corresponding sets of mappings would be the same (minor modifications are needed for the rest of the reduction as well). We proceed as follows. First, we may assume that Ψ_1 and Ψ_2 use the same variables $u_1, \ldots, u_n$ (ordered in the same way). Then, we can translate each Ψ_k, $k = 1, 2$, to an equivalent propositional formula Ψ'_k in 4CNF over the same variables such that for each three literals ℓ_1, ℓ_2, ℓ_3 over $u_1, \ldots, u_n$ (possibly with repetitions) Ψ'_k has a clause $\ell_1 \vee \ell_2 \vee \ell_3 \vee \ell'_1$, where ℓ'_1 is ℓ_1 if $\ell_1 \vee \ell_2 \vee \ell_3$ is in Ψ_k and the negation of ℓ_1 otherwise. Since the conjunctions $\mathsf{Cl}_\gamma^{\Psi_k}(z_\gamma) \wedge \bigwedge_{j=1}^{3} \mathsf{Arg}_j(z_\gamma, z_{u_\gamma^j})$ depend only on the variables in literals and not on their signs, the sets of mappings in $\mathcal{M}_{\Psi'_1}$ and $\mathcal{M}_{\Psi'_2}$ (constructed in the same way as $\mathcal{M}_{\Psi_k}$ except that j ranges over 1 to 4 instead of 1 to 3), differ only in the superscripts of relational names $\mathsf{Cl}_\gamma^{\Psi_k}$. Therefore we can use the same relational names Cl_γ instead of $\mathsf{Cl}_\gamma^{\Psi_1}$ and $\mathsf{Cl}_\gamma^{\Psi_2}$ in the instances and mappings, and then the union of $\mathcal{M}_{\Psi'_1}$ and $\mathcal{M}_{\Psi'_2}$ is just the set of CQ views $\mathcal{M}_{\Psi'_1}$. □

6. Policy Compliance and Strong Compliance

We now turn our attention to the policy compliance problems and show that both COMPLY and STRCOMPLY are decidable for any ontology language with decidable query entailment problem. Furthermore, we establish matching complexity bounds for both problems in the most common cases. In fact, the complexity of both problems coincide for all the studied cases, and the proof techniques required are also very similar.

6.1. Decidability and Complexity Upper Bounds

Consider an arbitrary input $(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ to COMPLY (or to STRCOMPLY). By Definition 5, a correct procedure must return `true` if and only if, for every tuple $\mathbf{a}$, there exists a source instance $\mathcal{D}_\mathbf{a}$ indistinguishable from $\mathcal{D}$ such that $\mathcal{D} \models p(\mathbf{a})$ implies $\mathcal{D}_\mathbf{a} \not\models p(\mathbf{a})$ (respectively, such that $\mathcal{D} \models \mathbf{a}$ if and only if $\mathcal{D}_\mathbf{a} \not\models p(\mathbf{a})$).

We start our discussion of a decision procedure with two basic observations. First, for a source instance $\mathcal{D}'$ to be indistinguishable from $\mathcal{D}$, the virtual image of $\mathcal{D}'$ via the mappings can only contain constants from $ADom(\mathcal{D})$. Second, the virtual image of $\mathcal{D}'$ is fully determined by the way in which the body of the mappings matches homomorphically to $\mathcal{D}'$ and, in particular, on the instantiation of the frontier variables of the mappings in such homomorphisms. These observations motivate the following definition of a *source type*.

Definition 16. *A* source type τ *for a data integration setting* $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ *is a function assigning* `true` *or* `false` *to each sentence of the form* $\exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$*, with* $\mathbf{a}$ *a tuple of constants from* ADom$(\mathcal{D})$ *and* $\varphi(\mathbf{x}, \mathbf{z})$ *the body of a mapping in* $\mathcal{M}$*. The* image *of* τ*, denoted* $\mathcal{V}_\tau$*, is the set of sentences* $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ *such that* $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ *is a mapping in* $\mathcal{M}$ *and* τ *returns* `true` *when applied to* $\exists \mathbf{z}.\, \varphi(\mathbf{a}, \mathbf{z})$*.*

Intuitively, each $\mathcal{V}_\tau$ associated to a source type τ represents a candidate virtual image. To decide compliance and strong compliance we are interested only in those source types that are *realisable* and *strongly realisable*, respectively. Following Definition 5, a source type τ is realisable if it has a witnessing source instance $\mathcal{D}_\tau$ that refutes some answer to the policy. Analogously, following Definition 6, τ is strongly realisable if it has a witnessing $\mathcal{D}_\tau$ such that the evaluation of the policy over $\mathcal{D}_\tau$ disagrees with that over $\mathcal{D}$.

Definition 17. *Let $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ be a data integration setting and p be a Boolean policy. A source type τ for $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ is*

- *p-*realisable *if there exists a source instance $\mathcal{D}_\tau$ such that $\mathcal{V}_{\mathcal{M},\mathcal{D}_\tau} = \mathcal{V}_\tau$ and $\mathcal{D}_\tau \not\models p$; and*
- strongly *p-*realisable *if there exists $\mathcal{D}_\tau$ such that $\mathcal{V}_{\mathcal{M},\mathcal{D}_\tau} = \mathcal{V}_\tau$ and $\mathcal{D}_\tau \models p$ if and only if $\mathcal{D} \not\models p$.*

Note that the notions of realisability in Definition 17 depend on the mappings $\mathcal{M}$ and the source instance $\mathcal{D}$, but are independent from the ontology $\mathcal{O}$. Indeed, the intention behind realisability is to capture the different ways in which the body of the mappings match the source data, and this is independent from the contents of the ontology.

The following lemma shows that realisability in both its flavours can be characterised as a satisfiability problem over a logical theory consisting of a Boolean combination of existentially quantified sentences. As a result, whenever the aforementioned logical theory is satisfiable, it has a model of polynomial size that can be guessed by a non-deterministic algorithm.

Lemma 18. *Let τ be a source type for a data integration setting $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ and ρ_τ be the conjunction of the following sentences:*

$$\begin{aligned} \xi, &\quad \textit{for all } \xi \textit{ with } \tau(\xi) = \texttt{true}, \\ \neg\xi, &\quad \textit{for all } \xi \textit{ with } \tau(\xi) = \texttt{false}, \\ \forall \mathbf{x}, \mathbf{z}.\, (\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \textstyle\bigwedge_{x \in \mathbf{x}} \bigvee_{a \in \mathrm{ADom}(\mathcal{D})} x = a), &\quad \textit{for all mappings in } \mathcal{M} \textit{ with body } \varphi(\mathbf{x}, \mathbf{z}). \end{aligned}$$

Then, given a Boolean policy p, τ is

- *p-realisable if and only if $\rho_\tau \wedge \neg p$ is satisfiable; and*
- *strongly p-realisable if and only if either $\mathcal{D} \models p$ and $\rho_\tau \wedge \neg p$ is satisfiable, or $\mathcal{D} \not\models p$ and $\rho_\tau \wedge p$ is satisfiable.*

Proof. We prove the statement in the lemma concerning p-realisability. The proof for the statement about strong p-realisability is analogous. Denote for the proof $\sigma = \rho_\tau \wedge \neg p$.

Assume that σ is satisfiable. Observe that p and each ξ in ρ_τ are existentially quantified sentences; furthermore, a formula $\forall \mathbf{x}, \mathbf{z}.\, (\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \bigwedge_{x \in \mathbf{x}} \bigvee_{a \in ADom(\mathcal{D})} x = a)$ is equivalent to $\neg \exists \mathbf{x}, \mathbf{z}.\, (\varphi(\mathbf{x}, \mathbf{z}) \wedge \bigvee_{x \in \mathbf{x}} \bigwedge_{a \in ADom(\mathcal{D})} x \neq a)$. Thus, σ is equivalent to a Boolean combination of existentially quantified sentences and, as a result, a satisfiable σ must have a polynomial size Herbrand model I that satisfies each ξ such that $\tau(\xi) = \texttt{true}$ by mapping each existentially quantified variable to a constant in $ADom(\mathcal{D})$. Such model corresponds directly to the source instance $\mathcal{D}_\tau$ required by Definition 17.

Conversely, if τ is p-realisable, then a source instance $\mathcal{D}_\tau$ satisfying the conditions of Definition 17 must exist. Such $\mathcal{D}_\tau$ satisfies $\neg p$ by definition, as well as every formula in ρ_τ by the requirement that $\mathcal{V}_{\mathcal{M},\mathcal{D}_\tau} = \mathcal{V}_\tau$. Hence, σ is satisfiable. □

Lemma 18 gives us a way to find a candidate for a witness for compliance and strong compliance if the policy is Boolean. For general policies $p(\mathbf{x})$ we need to find such a witness for every $p(\mathbf{a})$, where $\mathbf{a}$ is a tuple of constants in $ADom(\mathcal{D})$. Of course, by Lemma 3 in the previous section, a realisable type τ must satisfy an additional property to witness (strong) compliance: $\mathcal{O} \cup \mathcal{V}_\tau$ must be equivalent to $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$.

With these ingredients, we are ready to present alternating algorithms for Comply$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ and StrComply$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ for a data integration setting $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ and a policy p. The first one proceeds according to the following steps:

1. universally guess a tuple $\mathbf{a}$ of constants from $ADom(\mathcal{D})$ of the same arity as p;
2. existentially guess a source type τ for $(\mathcal{O}, \mathcal{M}, \mathcal{D})$;
3. verify whether
   - τ is $p(\mathbf{a})$-realisable, and
   - $\mathcal{O} \cup \mathcal{V}_\tau$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ are equivalent;
4. accept if both hold and reject otherwise.

The procedure for checking StrComply$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ is analogous, with the only difference that it verifies strong $p(\mathbf{a})$-realisability instead of $p(\mathbf{a})$-realisability.

Correctness of these algorithms follows from Lemma 3 and the definition of (strong) realisable type. Furthermore, by analysing these algorithms (and variants thereof), as well as by exploiting the logical characterisation of realisability in Lemma 18, we can obtain decidability and complexity upper bounds for a wide range of practically relevant cases.

Theorem 19. *Problems* COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *and* STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *for* $\mathcal{O}$ *in an ontology language* $\mathbb{O}$, $\mathcal{M}$ *in a mapping language* $\mathbb{M}$, *and* p *in a policy language* $\mathbb{P}$ *are in*

(1) NEXPTIME, *if*

- *(a)* $\mathbb{O}$ *has CQ entailment in* NEXPTIME *with* NP *data complexity, or*
- *(b)* $\mathbb{O}$ *has fact entailment in* NEXPTIME *with* NP *data complexity, and* $\mathbb{M}$ *is GAV;*

(2) Π_3^p, *if*

- *(a)* $\mathbb{O}$ *has CQ entailment in* NP, *and* $\mathbb{M}$ *has bounded frontier, or*
- *(b)* $\mathbb{O}$ *has fact entailment in* NP, *and* $\mathbb{M}$ *is GAV and has bounded frontier; and*

(3) Σ_2^p, *if the condition of Case (2) holds, and* $\mathbb{P}$ *has bounded arity.*

In data complexity, COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *and* STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *are in*

(4) NP, *if*

- *(a)* $\mathbb{O}$ *has CQ entailment in* NP *in data complexity, or*
- *(b)* $\mathbb{O}$ *has fact entailment in* NP *in data complexity, and* $\mathbb{M}$ *is GAV.*

Note that, in particular, the Π_3^p bound in combined complexity and the NP bound in data are applicable to OBDA settings where ontologies are DL-Lite$_\mathcal{R}$ and mappings are GAV. Also, *Cases (2)* and *(3)* can be relativised in the same way as *Cases (2)* and *(3)* of Theorem 12. For example, if $\mathbb{O}$ has CQ (or fact) entailment in a complexity class C with $\Sigma_2^p \subseteq C$ and all other conditions of *Case (2a)* (or *(2b)*, respectively) of Theorem 19 hold, then COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) and STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) are both in co$\mathrm{NP}^C$; in particular, if CQ (or fact) entailment is in EXPTIME, as for full TGDs, or in PSPACE, as for linear TGDs, then COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) and STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) are also in EXPTIME or PSPACE, respectively. *Cases (1)* and *(4)* can be relativised as well, but we need stronger closure properties of the complexity class for entailment. Therefore, we just note that *Case (1)* holds with the same proof if NEXPTIME is replaced with any of the usual complexity class subsuming NEXPTIME, such as EXPSPACE, while *Case (4)* holds if NP is replaced with PSPACE, NEXPTIME, etc. Also, in a similar way, we can prove decidability of our compliance problems (both in combined or data complexity) whenever the corresponding entailment problems for the ontology language at hand are decidable.

Proof. All the cases follow from the analysis of the algorithm presented before the theorem. We concentrate on problem COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$), and the argument for STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) is analogous.

In *Case (1)* there is no need to guess a tuple $\mathbf{a}$ of constants, because there are only exponential number of them, and we can go through them one by one deterministically. Then, any source type τ is of exponential size (same as $\mathcal{V}_\tau$), so we can guess it in NEXPTIME. Checking for $p(\mathbf{a})$-realisability of τ can also be done in NEXPTIME: first we guess an interpretation of exponential size and then verify that it is a model of ρ_τ and $\neg p(\mathbf{a})$. Note that the verification can be done in exponential time in the size of the input, because each existentially quantified subformula of $\rho_\tau \wedge \neg p(\mathbf{a})$ is of polynomial size. Finally, checking for equivalence of $\mathcal{O} \cup \mathcal{V}_\tau$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ can also be done in NEXPTIME by checking exponential number of CQ or fact entailments (in *Case (1a)* or *(1b)*, respectively), which are in NEXPTIME with NP data complexity.

In *Case (2)* the mappings in $\mathcal{M}$ have bounded frontier. Therefore, any source type for $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ is of polynomial size. So, we can reformulate the algorithm presented before the theorem as follows in order to directly justify the Π_3^p upper bound:

1. universally guess a tuple $\mathbf{a}$ of constants;
2. existentially guess
   - a source type τ,
   - a source interpretation I and homomorphisms witnessing that I is a model of the positive elements of ρ_τ,
   - witnesses for entailment of all elements $\mathcal{V}_\tau$ by $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ and of $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ by $\mathcal{O} \cup \mathcal{V}_\tau$,
3. universally guess non-witnesses that I is a model of the negative elements of ρ_τ and $\neg p(\mathbf{a})$;
4. check that the witnesses indeed satisfy the required properties—that is, τ is $p(\mathbf{a})$-realisable, and $\mathcal{O} \cup \mathcal{V}_\tau$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ are equivalent.

Note that in *Case (2a)* we use CQ entailment, while in *Case (2b)* we resort to fact entailment.

For *Case (3)* we can use almost the same variant of the algorithm as for *Case (2)*. The only difference is that instead of guessing on the first step we can go through all the tuples $\mathbf{a}$ one by one, because their size—that is, the arity of p—is bounded.

Case (4) again requires a very similar variant of the algorithm as the previous two cases. However, since $\mathcal{M}$, $\mathcal{O}$, and p are fixed, we only need to guess I and witnesses for entailments. □

Similarly to indistinguishability, compliance can be done more efficiently if the ontology is linear and mappings are LAV. This is justified by the following lemma, which claims that we can restrict ourselves to source instances of polynomial size.

Lemma 20. *Let $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ be a data integration setting with linear $\mathcal{O}$ and LAV $\mathcal{M}$, and let p be a Boolean policy. If there exists a p-realisable (or strongly p-realisable) source type τ for $(\mathcal{O}, \mathcal{M}, \mathcal{D})$ such that $\mathcal{O} \cup \mathcal{V}_\tau$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ are equivalent, then there exists a source type with the same properties that is* `true` *on at most $|\mathcal{M}|^2 \cdot |\mathcal{D}| \cdot k_{\max}$ sentences, where $k_{\max}$ is the maximal number of atoms in heads of mappings in $\mathcal{M}$.*

Proof. The proof is based on Lemma 11. Again, we concentrate of realisability, and the case of strong realisability is similar.

Since τ is p-realisable, there exists a source instance $\mathcal{D}_\tau$ such that $\mathcal{V}_{\mathcal{M},\mathcal{D}_\tau} = \mathcal{V}_\tau$ and $\mathcal{D}_\tau \not\models p$. Then, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_\tau} \models \mathcal{V}_{\mathcal{M},\mathcal{D}}$, and, by Lemma 11, for each mapping $\mathsf{R}(\mathbf{x}, \mathbf{z}) \to \exists \mathbf{y}.\, \psi(\mathbf{x}, \mathbf{y})$ in $\mathcal{M}$ and tuple of constants $\mathbf{a}, \mathbf{c}$ with $\mathsf{R}(\mathbf{a}, \mathbf{c}) \in \mathcal{D}$ there are an integer k bounded by the number $|\psi|$ of atoms in ψ, mappings $\mathsf{R}_i(\mathbf{x}_i, \mathbf{z}_i) \to \exists \mathbf{y}_i.\, \psi_i(\mathbf{x}_i, \mathbf{y}_i)$ in $\mathcal{M}$ and tuples of constants $\mathbf{a}_i, \mathbf{c}_i$, for $i = 1, \dots, k$, such that $\mathsf{R}_i(\mathbf{a}_i, \mathbf{c}_i) \in \mathcal{D}_\tau$ for all i and $\mathcal{O} \cup \{\exists \mathbf{y}_i.\, \psi_i(\mathbf{a}_i, \mathbf{y}_i) \mid i = 1, \dots, k\} \models \exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$. Consider the sub-instance $\mathcal{D}'_\tau$ of $\mathcal{D}_\tau$ that consists of all those $\mathsf{R}_i(\mathbf{a}_i, \mathbf{c}_i)$. On the one hand, by construction, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'_\tau}$ and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ are equivalent, as well as $\mathcal{D}'_\tau \not\models p$. On the other, again by construction, $\mathcal{D}'_\tau$ consists of at most $|\mathcal{M}| \cdot |\mathcal{D}| \cdot k_{\max}$ facts. Hence, the source type τ' that is `true` on $\exists \mathbf{z}.\, \mathsf{R}(\mathbf{a}, \mathbf{z})$ if and only if there is $\mathbf{c}$ with $\mathsf{R}(\mathbf{a}, \mathbf{c}) \in \mathcal{D}'_\tau$ satisfies all the properties required in the claim of the lemma. □

This lemma gives us a possibility to optimise the algorithm from Theorem 19 and obtain better upper bounds in the case when the ontology is linear and mappings are LAV.

Theorem 21. *Problems* COMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *and* STRCOMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *for $\mathcal{O}$ in an ontology language $\mathbb{O}$, $\mathcal{M}$ in a mapping language $\mathbb{M}$, and p in a policy language $\mathbb{P}$ are in*

(1) Π^p_3*, if*

- *(a) $\mathbb{O}$ is linear and has CQ entailment in* NP*, and $\mathbb{M}$ is LAV, or*
- *(b) $\mathbb{O}$ is linear and has fact entailment in* NP*, and $\mathbb{M}$ is GAV and LAV;*

(2) Σ^p_2*, if the condition of Case (1) holds, and $\mathbb{P}$ is of bounded arity; and*

(3) NP*, if the condition of Case (1) holds, and $\mathbb{P}$ is ground.*

Note that the best upper bound we can get from Theorem 19 for all three cases is NEXPTIME, and we will prove the matching lower bound for non-LAV mappings and even empty ontologies in Theorem 24. Again, the *Cases (1)* and *(2)* can be relativised as usual, while NP in *Case (3)* can be replaced with the standard complexity classes subsuming NP.

Proof. For *Case (1)* we can use almost the same variant of the algorithm as for *Case (2)* of Theorem 19: we just restrict ourselves to source instances τ and interpretations I of polynomial size, which is enough by Lemmas 18 and 20.

Case (2) differs from *Case (1)* in the same way as *Case (3)* of Theorem 19 differs from its *Case (2)*: there is no need to guess $\mathbf{a}$ because the arity of p is bounded.

For *Case (3)* we analyse the algorithm closer. Since $\mathcal{M}$ contains only LAV mappings, there is no need to guess witnesses that the interpretation I is a model for elements of ρ_τ, both positive and negative. Moreover, even if the domain of τ can be of exponential size, we need to verify its value on only a polynomial number of sentences. In particular, we can just check which mappings in $\mathcal{M}$ match which facts holding in I, and this can be done in polynomial time. Because of this and the fact that p is ground, there is no need for the second universal guess, and the problem can be solved with a polynomial existential guess. □

As we will see in the next section, the upper bounds established in Theorems 19 and 21 are tight. These bounds, in particular those concerning data complexity, can be seen as rather discouraging for practical applications. Next we address these concerns to the extent possible by identifying practically relevant islands of tractability for policy compliance and strong compliance. In particular, we show that if the ontology language has tractable fact entailment (as is the case for DL-Lite$_\mathcal{R}$), the mappings are GAV and LAV, policies are ground, and the arity of the source schema is bounded, then the problems are also tractable even in combined complexity. If we consider data complexity, then we can drop the requirement of bounded source arity, because it is implicit. Note that all of the aforementioned assumptions are required to obtain tractability: as we will show in the next section, any relaxation leads to NP-hardness of compliance and strong compliance.

Thus, we next present high-level algorithms that decide COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) and STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) under the aforementioned assumptions. For COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$), the algorithm proceeds according to the following steps, where $\mathbf{C}$ is a set of $s_{\max}$ distinct fresh constants with $s_{\max}$ the maximal source arity of $\mathcal{M}$:

1. construct $\mathcal{V}_{\mathcal{M},\mathcal{D}}$;
2. for each fact γ in p
   - construct $\mathcal{D}_\gamma$ as the set of all facts α over constants $ADom(\mathcal{D}) \cup \mathbf{C}$ such that
     - $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \mathcal{V}_{\mathcal{M},\{\alpha\}}$, and
     - $\alpha \neq \gamma$;
   - return `true` if $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_\gamma} \models \mathcal{V}_{\mathcal{M},\mathcal{D}}$;
3. return `false`.

The algorithm attempts to construct a source instance witnessing compliance. Each such instance should miss at least one fact from p, so the algorithm iterates through all such facts γ, checking whether the maximal set $\mathcal{D}_\gamma$ of facts that does not include γ and does not violate the virtual image of $\mathcal{D}$ is indistinguishable from $\mathcal{D}$. Such $\mathcal{D}_\gamma$ for any γ is a witness for compliance. Moreover, since the mappings are LAV, we can restrict the facts in the pool to only those that use constants $ADom(\mathcal{D}) \cup \mathbf{C}$—that is, we only need a few constants that do not propagate through the mappings.

Note that the aforementioned algorithm can be easily modified to decide STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) under the same assumptions. The modification first checks whether $\mathcal{D} \models p$; if it holds, then the procedure for COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) is evaluated; otherwise, `true` is returned if and only if $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}} \models \mathcal{V}_{\mathcal{M},\mathcal{D}'}$, where $\mathcal{D}'$ is the extension of $\mathcal{D}$ with all the facts in p.

The tractability results in the following theorem are justified by the correctness of our algorithms and the analysis of their running time.

Theorem 22. *Problems* COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *and* STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *for $\mathcal{O}$ in an ontology language $\mathbb{O}$, $\mathcal{M}$ in a mapping language $\mathbb{M}$, and p in a policy language $\mathbb{P}$ are in* AC^0 *if*

(a) $\mathbb{O}$ has CQ entailment in AC^0*, $\mathbb{M}$ is LAV and of bounded source and global arities, and $\mathbb{P}$ is ground; or*

(b) $\mathbb{O}$ has fact entailment in AC^0*, $\mathbb{M}$ is GAV, LAV and of bounded source and global arities, and $\mathbb{P}$ is ground.*

The same holds if all the problems are considered in data complexity (in which case the boundedness of the arities is automatic).

Note that in case of combined complexity the conditions for $\mathcal{M}$ imply bounded frontier. All cases except *Case (a)* for combined complexity are applicable to the settings with empty ontology, and the data complexity bounds is applicable to the settings with DL-Lite$_\mathcal{R}$ ontologies. As usual, in all cases AC^0 can be replaced with any of the standard complexity classes, such as NLOGSPACE, P and NP. In particular, in combined complexity, CQ entailment is EXPTIME-, PSPACE-, NP-, and NP-complete for full TGDs, linear TDGs, DL-Lite$_\mathcal{R}$, and empty ontologies, respectively, so *Case (a)* of the theorem implies that both COMPLY and STRCOMPLY are in EXPTIME, PSPACE, NP, and NP, respectively, for the corresponding settings; next, fact entailment is EXPTIME-, PSPACE-, and NLOGSPACE-complete for full TGDs, linear TDGs, and DL-Lite$_\mathcal{R}$, respectively, so *Case (b)* implies that both problems are in EXPTIME, PSPACE, and NLOGSPACE, respectively; finally, in data complexity, CQ and fact entailment are EXPTIME- and PSPACE-complete for full and linear TDGs, respectively, so the data complexity results imply that COMPLY and STRCOMPLY are in EXPTIME and PSPACE, respectively.

Proof. The proofs are based on the algorithms provided before the theorem. We start with compliance.

We first claim that the algorithm is sound and complete—that is, it returns `true` if and only if COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) is `true`. The forward direction follows immediately from Lemma 3 and the fact that, by construction, the algorithm found a source instance $\mathcal{D}_\gamma$ such that $\mathcal{D}_\gamma \not\models p$, and $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}}$ is equivalent to $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_\gamma}$. Next we concentrate on backward direction. By Lemma 3, it is enough to show that if there exists a source instance $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ with $\mathcal{D}' \not\models p$ then there exists a fact γ in p such that $\mathcal{D}_\gamma$ is indistinguishable from $\mathcal{D}'$ (and hence from $\mathcal{D}$). Note that, by construction, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}'} \models \mathcal{V}_{\mathcal{M},\mathcal{D}_\gamma}$ so we only argue that $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_\gamma} \models \mathcal{V}_{\mathcal{M},\mathcal{D}'}$. Let $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y})$ be any CQ in $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$, and let α' be a fact in $\mathcal{D}'$ such that $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y}) \in \mathcal{V}_{\mathcal{M},\{\alpha'\}}$—that is, the fact witnessing the CQ in $\mathcal{D}'$. Then a fact α obtained from α' by replacing all the constants not in $ADom(\mathcal{D})$ by (distinct) constants in $\mathbf{C}$ is such that $\mathcal{V}_{\mathcal{M},\{\alpha\}} = \mathcal{V}_{\mathcal{M},\{\alpha'\}}$; in particular, $\exists \mathbf{y}.\, \psi(\mathbf{a}, \mathbf{y}) \in \mathcal{V}_{\mathcal{M},\{\alpha\}}$. Moreover, since $\alpha' \neq \gamma$, $\alpha \neq \gamma$ as well. Therefore, since $\mathcal{V}_{\mathcal{M},\mathcal{D}'} = \mathcal{V}_{\mathcal{M},\mathcal{D}}$, we have that $\alpha \in \mathcal{D}_\gamma$. So, overall, $\mathcal{V}_{\mathcal{M},\mathcal{D}'} \subseteq \mathcal{V}_{\mathcal{M},\mathcal{D}_\gamma}$—that is, $\mathcal{O} \cup \mathcal{V}_{\mathcal{M},\mathcal{D}_\gamma} \models \mathcal{V}_{\mathcal{M},\mathcal{D}'}$ as required for the correctness.

Finally, we argue that our algorithm is feasible in AC^0 under the conditions of the theorem (essentially, the same proof works for both cases and both for combined and data complexity). This is implied by the following observations:

1. the bound on the arity of the source schema allows us to construct all facts over this schema in AC^0 (recall that in the case of data complexity the bound is implicit);

2. the fact that the mappings are LAV implies that the virtual images of instances can also be constructed in AC^0;

3. the complexity bound on CQ or fact entailment implies that the checks required for each fact α as well as the entailment in the last step of the cycle are all feasible in AC^0.

This concludes the proof for the compliance problem. The case of strong compliance is analogous when $\mathcal{D} \models p$, and straightforward when $\mathcal{D} \not\models p$. □

6.2. Complexity Lower Bounds

An obvious drawback of our generic algorithm for (strong) compliance presented in the previous section is that, in many cases, there is a need for guessing a source type, which can be of exponential size. Unfortunately, as we show in this section, our generic algorithm cannot be improved in general: all the upper bounds in Section 6.1 are tight.

As we did in Section 5.2 for source indistinguishability, we begin with the observation that query entailment reduces to compliance checking under rather mild conditions on the mapping and policy languages. As a result, complexity lower bounds on standard entailment problems for ontologies easily transfer to compliance checking.

Proposition 23. *Problems* COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *and* STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *are* C*-hard, for a complexity class* C*, if* $\mathcal{O}$ *ranges over an ontology language* $\mathbb{O}$*,* $\mathcal{M}$ *over a mapping language* $\mathbb{M}$*, and* p *ranges over a policy language* $\mathbb{P}$ *such that*

(a) CQ entailment is C*-hard for* $\mathbb{O}$ *and* $\mathbb{M}$ *contains all sets of LAV mappings of source arity bounded by 0, or*

(b) fact entailment is C*-hard for* $\mathbb{O}$ *and* $\mathbb{M}$ *contains all sets of mappings that are GAV, LAV, and of source arity bounded by 0,*

and, in both cases, $\mathbb{P}$ *contains all facts; if, additionally, the arity of the schema of every ontology in* $\mathbb{O}$ *is bounded by a number k, then the required sets of mappings in* $\mathbb{M}$ *may be additionally restricted to have the global arity bounded by k. The same holds if the problems are considered in data complexity (in which case the boundedness of the arities are automatic).*

Recall that in this setting the bound on the source arity implies the same bound on the frontier. Note also that, in combined complexity, CQ entailment is NP-complete even for empty ontologies, so this proposition provides a matching lower bound for *Case (3)* of Theorem 21 (another, incomparable lower bound for this case will be obtained in Theorem 28). It also provides matching lower bounds to *Cases (2a)–(3b)* of Theorem 19, *Cases (1a)–(3b)* of Theorem 21 and *Cases (a)* and *(b)* of Theorem 22 for settings with non-empty ontologies (e.g., full TGDs, linear TGDs, and DL-Lite$_\mathcal{R}$) when CQ or fact entailment dominates the complexity of other steps of the corresponding algorithms.

Proof. We again concentrate on COMPLY, and the proof for STRCOMPLY is analogous.

For *Case (a)*, consider an input to the CQ entailment problem consisting of an ontology $\mathcal{O}_0$, an instance $\mathcal{D}_0$, and a Boolean CQ q_0. We construct a set of LAV mappings $\mathcal{M}$ with source arity bounded by 0, a source instance $\mathcal{D}$, and a fact α such that $\mathcal{O}_0, \mathcal{D}_0 \models q_0$ if and only if COMPLY($\mathcal{O}_0, \mathcal{M}, \mathcal{D}, \alpha$) is `true`.

Let the source schema consists of two nullary relational names, D and P, the source instance $\mathcal{D}$ consist of two facts $\mathsf{D}()$ and $\mathsf{P}()$, and the policy fact α be $\mathsf{P}()$.

Let also the global schema consist of all relational names in $\mathcal{D}_0$, while the set $\mathcal{M}$ of GAV and LAV mappings consist of

$$\begin{array}{lcll} \mathsf{D}() & \rightarrow & \beta, & \text{for each fact } \beta \text{ in } \mathcal{D}_0, \text{ and} \\ \mathsf{P}() & \rightarrow & q_0. & \end{array}$$

Note that $\mathcal{V}_{\mathcal{O},\{\mathsf{D}()\}} = \mathcal{D}_0$, so it is immediate to check that

1. if $\mathcal{O}_0, \mathcal{D}_0 \models q_0$ then the source instance $\{\mathsf{D}()\}$ witnesses COMPLY($\mathcal{O}_0, \mathcal{M}, \mathcal{D}, \alpha$), and

2. if COMPLY($\mathcal{O}_0, \mathcal{M}, \mathcal{D}, \alpha$) then the witnessing source instance, which is either $\emptyset$ or $\{\mathsf{D}()\}$, guarantees that $\mathcal{O}_0, \mathcal{D}_0 \models q_0$.

Case (b) can be proved in the same way: the only difference is that q_0 is a fact, so $\mathcal{M}$ is GAV.

The same proofs work for the statements about ontologies of bounded arity and data complexity. □

In the rest of this section we give matching lower bounds for different versions of the compliance and strong compliance problems for the cases when entailment checking is not the dominating source of complexity.

Recall that, according to *Case (1)* of Theorem 19, in the most general settings for empty ontologies compliance and strong compliance are in NEXPTIME. We start by showing two rather strong incomparable matching lower bounds: the first holds already if we restrict the input ontology to be empty, the mappings to be of bounded source and global arities, and the policy

to be just a fact, while the second additionally restricts the mappings to be CQ views (and hence GAV), but does not bound the global arity.

Our proof uses an encoding of the well-known NEXPTIME-complete version of the domino tiling problem. In the source, there are relational names associating "cell objects" with vertical and horizontal coordinates, and also with tile types. The only information exported by the mappings is that adjacent coordinates are associated with some cells and with some tile type assignments which are compatible. In fact, in the input source instance $\mathcal{D}$, a cell will be associated with every tile type, since there is only one cell object. But this information is not exported by the mappings, and thus source instances indistinguishable from $\mathcal{D}$ may be better behaved. The policy p is chosen so that indistinguishable source instances where p fails correspond to those where coordinates are assigned a unique tiling type.

Theorem 24. *The problems* COMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *and* STRCOMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *are* NEXPTIME*-hard if* $\mathcal{O}$ *is empty,* $\mathcal{M}$ *ranges over a mapping language* $\mathbb{M}$*, and* p *ranges over a policy language* $\mathbb{P}$ *such that either*

(a) $\mathbb{M}$ *contains all sets of mappings with source and global arity bounded by 2, or*

(b) $\mathbb{M}$ *contains all sets of CQ views with source arity bounded by 2,*

and, in both cases, $\mathbb{P}$ *contains all facts.*

As mentioned before, this theorem gives two incomparable matching lower bounds for *Case (1)* of Theorem 19. By Theorems 19 and 21, these bounds cannot be improved along the dimensions considered in this paper; for example, if we join the requirements of *Cases (a)* and *(b)*, then the problem is in Σ_2^p by *Case (3)* of Theorem 19 (or, alternatively, by *Case (2)* of Theorem 21).

Proof. We show the NEXPTIME lower bounds stated in the theorem by means of reductions of a corresponding variation of the tiling problem. The input of this problem is a *tiling instance* $(\mathcal{T}, R_H, R_V)$, with a finite set of tile types $\mathcal{T}$, horizontal compatibility relation $R_H \subseteq \mathcal{T} \times \mathcal{T}$ and vertical compatibility relation $R_V \subseteq \mathcal{T} \times \mathcal{T}$, while the answer is `true`—that is, the instance *has a solution*—if and only if it is possible to tile a $2^n \times 2^n$ square, for $n = |\mathcal{T}|$, according to R_H and R_V.

We first describe the reduction and prove its correctness for *Case (b)*, and then explain how to adapt it to *Case (a)*.

Let $(\mathcal{T}, R_H, R_V)$ be a tiling instance, and let $n = |\mathcal{T}|$. We will first show how to construct a set of GAV mappings $\mathcal{M}$, a Boolean UCQ p and a source instance $\mathcal{D}$ such that COMPLY$(\emptyset, \mathcal{M}, \mathcal{D}, p) =$ `true` if and only if the instance has a solution. Afterwards, we discuss how this construction can be modified to make the set of mappings consist of CQ views and the policy be a single Boolean CQ. To complete the proof for *Case (b)*, we show how, with a minor modification of the previous construction, we can further restrict p to be a fact.

It is important to note that $\mathcal{D}$ and p we will construct are such that $\mathcal{D} \models p$. In this case the notions of compliance and strong compliance coincide, and therefore the reductions show NEXPTIME-hardness of STRCOMPLY as well.

We start with the definition of the source schema: let it consist of unary relational names Zero, One and Tiled_t for each $t \in \mathcal{T}$, as well as binary relational names HBit_i and VBit_i for each i such that $n \geq i \geq 1$.

Then, let source instance $\mathcal{D}$ consist of the following facts over constants 0, 1, and e:

$$\begin{array}{ll} \mathsf{Zero}(0), \mathsf{One}(1), & \\ \mathsf{Tiled}_t(e), & \text{for } t \in \mathcal{T}, \\ \mathsf{HBit}_i(e,0), \mathsf{HBit}_i(e,1), & \text{for } n \geq i \geq 1, \text{ and} \\ \mathsf{VBit}_i(e,0), \mathsf{VBit}_i(e,1), & \text{for } n \geq i \geq 1. \end{array}$$

Let now the target schema consists of relational names HValid_i and VValid_i, with $n \geq i \geq 1$, all with arity $3n$.

We next construct $\mathcal{M}$. For the sake of readability, we allow for (safe) equalities of variables in the bodies of mappings, which clearly does not add any power to the language and can be eliminated by a polynomial transformation. We also introduce some abbreviations. First, for tuples of variables $\mathbf{u} = u_n, \ldots, u_1$ and $\mathbf{v} = v_n, \ldots, v_1$, let

$$\begin{aligned} &\mathit{Next}_1(\mathbf{u}, \mathbf{v}) = (u_n = v_n) \wedge \cdots \wedge (u_2 = v_2) \wedge \mathsf{Zero}(u_1) \wedge \mathsf{One}(v_1), \\ &\quad \cdots \\ &\mathit{Next}_i(\mathbf{u}, \mathbf{v}) = (u_n = v_n) \wedge \cdots \wedge (u_{i+1} = v_{i+1}) \wedge \mathsf{Zero}(u_i) \wedge \mathsf{One}(v_i) \wedge \mathsf{One}(u_{i-1}) \wedge \mathsf{Zero}(v_{i-1}) \wedge \cdots \wedge \mathsf{One}(u_1) \wedge \mathsf{Zero}(v_1), \\ &\quad \cdots \\ &\mathit{Next}_n(\mathbf{u}, \mathbf{v}) = \mathsf{Zero}(u_n) \wedge \mathsf{One}(v_n) \wedge \mathsf{One}(u_{n-1}) \wedge \mathsf{Zero}(v_{n-1}) \wedge \cdots \wedge \mathsf{One}(u_1) \wedge \mathsf{Zero}(v_1). \end{aligned}$$

Intuitively, $\mathit{Next}_i(\mathbf{u}, \mathbf{v})$ is `true` whenever $\mathbf{u}$ and $\mathbf{v}$ represent in binary consecutive numbers of the form $b_n \ldots b_{i+1} 01 \ldots 1$ and $b_n \ldots b_{i+1} 10 \ldots 0$, respectively, with $b_j \in \{0, 1\}$ for $n \leq j < i$. Also, let, for $\mathbf{x} = x_n, \ldots, x_1$ and $\mathbf{y} = y_n, \ldots, y_1$,

$$\mathit{Coords}(z, \mathbf{x}, \mathbf{y}) = \mathsf{HBit}_n(z, x_n) \wedge \cdots \wedge \mathsf{HBit}_1(z, x_1) \wedge \mathsf{VBit}_n(z, y_n) \wedge \cdots \wedge \mathsf{VBit}_1(z, y_1).$$

We next exploit these abbreviations to define set of mappings $\mathcal{M}$. For each i, $n \geq i \geq 1$, and $(t_1, t_2) \in R_H$, let $\mathcal{M}$ include the following mapping, which checks that all horizontally adjacent cells are assigned to tile types in accordance with R_H:

$$\mathit{Next}_i(\mathbf{x}_1, \mathbf{x}_2) \wedge \mathit{Coords}(z_1, \mathbf{x}_1, \mathbf{y}) \wedge \mathit{Coords}(z_2, \mathbf{x}_2, \mathbf{y}) \wedge \mathsf{Tiled}_{t_1}(z_1) \wedge \mathsf{Tiled}_{t_2}(z_2) \rightarrow \mathsf{HValid}_i(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}).$$

Similarly, let $\mathcal{M}$ include, for each i, $n \geq i \geq 1$, and $(t_1, t_2) \in R_V$, the following mapping, which checks that all vertically adjacent cells are assigned according to R_V:

$$\mathit{Next}_i(\mathbf{y}_1, \mathbf{y}_2) \wedge \mathit{Coords}(z_1, \mathbf{x}, \mathbf{y}_1) \wedge \mathit{Coords}(z_2, \mathbf{x}, \mathbf{y}_2) \wedge \mathsf{Tiled}_{t_1}(z_1) \wedge \mathsf{Tiled}_{t_2}(z_2) \rightarrow \mathsf{VValid}_i(\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2).$$

Finally, let p be the following Boolean UCQ with all parameters implicitly existentially quantified variables:

$$\bigvee_{t_1, t_2 \in \mathcal{T},\ t_1 \neq t_2} \mathit{Coords}(z_1, \mathbf{x}, \mathbf{y}) \wedge \mathit{Coords}(z_2, \mathbf{x}, \mathbf{y}) \wedge \mathsf{Tiled}_{t_1}(z_1) \wedge \mathsf{Tiled}_{t_2}(z_2).$$

We claim that $(\emptyset, \mathcal{M}, \mathcal{D})$ complies to p if and only if the tiling instance $(\mathcal{T}, R_H, R_V)$ has a solution. To prove this in the forward direction, assume that there exists a source instance $\mathcal{D}'$ that is indistinguishable from $\mathcal{D}$ and satisfies $\neg p$. Since $\mathcal{D}$ and $\mathcal{D}'$ are indistinguishable, we know that for each pair of bit vectors $\mathbf{b}_x, \mathbf{b}_y$ there is at least one cell object c and tile type t such that $\mathit{Coords}(c, \mathbf{b}_x, \mathbf{b}_y)$ and $\mathsf{Tiled}_t(c)$ are in $\mathcal{D}'$. Further, the fact that $\neg p$ holds guarantees that there is at most one such t for c. We define a tiling by setting tile type t at coordinates $\mathbf{b}_x, \mathbf{b}_y$, and indistinguishability further guarantees that the compatibility relations are satisfied. Conversely, if the tiling instance has a solution, we can define a source instance $\mathcal{D}'$ by creating a unique c for each pair of bit vectors $\mathbf{b}_x, \mathbf{b}_y$, associating c with $\mathbf{b}_x, \mathbf{b}_y$ via *Coords*, and adding fact $\mathsf{Tiled}_t(c)$ only for the tile type t given to $\mathbf{b}_x, \mathbf{b}_y$ in the solution. It is easy to see that the resulting $\mathcal{D}'$ is indistinguishable from $\mathcal{D}$ and satisfies $\neg p$. This completes the proof of correctness of our reduction.

Observe, however, that the mappings $\mathcal{M}$ in our reduction are not CQ views since there is a mapping with head relational name HValid_i for every pair (t_1, t_2) of horizontally compatible types in R_H (and, similar mappings for VValid_i); moreover, the policy is a UCQ. Next, we show how to transform these disjunctions into conjunctions. The idea is to have tile types as objects in a new "type storage" relation, along with a relation storing compatibility.

Next we make this intuition formal. Instead of unary relational names Tiled_t, the source schema contains binary relational names TypeOf, $\mathsf{DiffTypes}$, HComp, and VComp. In turn, the global schema, besides HValid_i and VValid_i, contains the mirror binary relational names $\mathsf{DiffTypes}'$, HComp', and VComp'.

In turn, let the source instance $\mathcal{D}_1$ be the same as $\mathcal{D}$, except that instead of the facts $\mathsf{Tiled}_t(e)$, for $t \in \mathcal{T}$, it contains the following facts over constants e and d_t, $t \in \mathcal{T}$:

$$\begin{array}{ll} \mathsf{TypeOf}(e, d_t), & \text{for all } t \in \mathcal{T}, \\ \mathsf{DiffTypes}(d_t, d_{t'}), & \text{for all } t, t' \in \mathcal{T} \text{ with } t \neq t', \\ \mathsf{HComp}(d_{t_1}, d_{t_2}), & \text{for all } (t_1, t_2) \in R_H, \text{ and} \\ \mathsf{VComp}(d_{t_1}, d_{t_2}), & \text{for all } (t_1, t_2) \in R_V. \end{array}$$

Let $\mathcal{M}_1$ contain the mappings

$$\begin{array}{rcl} \mathsf{DiffTypes}(w, w') & \rightarrow & \mathsf{DiffTypes}'(w, w'), \\ \mathsf{HComp}(w_1, w_2) & \rightarrow & \mathsf{HComp}'(w_1, w_2), \\ \mathsf{VComp}(w_1, w_2) & \rightarrow & \mathsf{VComp}'(w_1, w_2), \end{array}$$

and, for each i with $n \geq i \geq 1$, the mappings

$$\mathit{Next}_i(\mathbf{x}_1, \mathbf{x}_2) \wedge \mathit{Coords}(z_1, \mathbf{x}_1, \mathbf{y}) \wedge \mathit{Coords}(z_2, \mathbf{x}_2, \mathbf{y}) \wedge \mathsf{TypeOf}(z_1, w_1) \wedge \mathsf{TypeOf}(z_2, w_2) \wedge \mathsf{HComp}(w_1, w_2) \rightarrow \mathsf{HValid}_i(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}),$$
$$\mathit{Next}_i(\mathbf{y}_1, \mathbf{y}_2) \wedge \mathit{Coords}(z_1, \mathbf{x}, \mathbf{y}_1) \wedge \mathit{Coords}(z_2, \mathbf{x}, \mathbf{y}_2) \wedge \mathsf{TypeOf}(z_1, w_1) \wedge \mathsf{TypeOf}(z_2, w_2) \wedge \mathsf{VComp}(w_1, w_2) \rightarrow \mathsf{VValid}_i(\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2).$$

Mappings $\mathcal{M}_1$ are CQ views as required. Finally, let p_1 be the following Boolean CQ, where all parameters are implicitly existentially quantified variables:

$$\mathit{Coords}(z_1, \mathbf{x}, \mathbf{y}) \wedge \mathit{Coords}(z_2, \mathbf{x}, \mathbf{y}) \wedge \mathsf{TypeOf}(z_1, w_1) \wedge \mathsf{TypeOf}(z_2, w_2) \wedge \mathsf{DiffTypes}(w_1, w_2).$$

We claim that $(\emptyset, \mathcal{M}_1, \mathcal{D}_1)$ complies to p_1 if and only if the tiling instance $(\mathcal{T}, R_H, R_V)$ has a solution. For the forward direction, assume that there is an indistinguishable from $\mathcal{D}_1$ source instance $\mathcal{D}'$ satisfying $\neg p_1$. As in the previous case, each coordinate pair is associated with a tile type in a way that horizontally and vertically adjacent cells have compatible types. Moreover, since $\mathcal{D}'$ satisfies $\neg p_1$, this type is unique for each cell. The other direction is also very similar to the previous case.

Next, we further modify the reduction to make the policy a fact. To this end, we extend the source schema with unary relational names Choice, Check and $\mathsf{Trigger}$, while the global schema with unary Choice' and Check' as well as nullary $\mathsf{Trigger}'$ and $\mathsf{HasTwoTypes}$.

Then, we define a source instance $\mathcal{D}_2$ as an extension of $\mathcal{D}_1$ with the following atoms, where d is a fresh constant:

$$\mathsf{Choice}(e), \mathsf{Choice}(d), \mathsf{Check}(e), \mathsf{Trigger}(d).$$

We also define $\mathcal{M}_2$ as an extension of $\mathcal{M}_1$ with the mappings

$$\begin{aligned} \mathsf{Choice}(u) &\rightarrow \mathsf{Choice}'(u), \\ \mathsf{Check}(u) &\rightarrow \mathsf{Check}'(u), \\ \mathsf{Trigger}(u) \wedge \mathsf{Choice}(u) &\rightarrow \mathsf{Trigger}'(), \end{aligned}$$

and the mapping

$$\mathsf{Check}(u) \wedge \mathsf{Trigger}(u) \wedge \beta \rightarrow \mathsf{HasTwoTypes}(),$$

where β is the body of p_1.

Finally, the policy p_2 is now the fact

$$\mathsf{Trigger}(d).$$

We claim that $(\emptyset, \mathcal{M}_2, \mathcal{D}_2)$ complies to p_2 if and only if the tiling instance $(\mathcal{T}, R_H, R_V)$ has a solution. Indeed, the virtual image of $\mathcal{M}_2$ and $\mathcal{D}_2$ contains the facts $\mathsf{Choice}'(e)$, $\mathsf{Choice}'(d)$, $\mathsf{Check}'(e)$, and $\mathsf{Trigger}'()$, but does not contain $\mathsf{HasTwoTypes}()$. So, any indistinguishable instance contains either $\mathsf{Trigger}(e)$ or $\mathsf{Trigger}(d)$. Since $\mathsf{Trigger}(d)$ is the policy, an indistinguishable instance witnessing compliance contains $\mathsf{Trigger}(e)$. Since it also contains $\mathsf{Check}(e)$, the last mapping is not applicable, which is required by indistinguishability, if and only if p_1 does not have a match in $\mathcal{D}_2$.

We next discuss how to adapt the reduction for *Case (b)* to *Case (a)*, where the target arity is bounded by 2 but mappings are not required to be CQ views (or even GAV). We can do it in the same way as in Theorem 14: instead of $3n$-ary global relational names HValid_i and VValid_i, we can use, for each i, $6n$ binary relational names HValid_i^j and VValid_i^j, for $1 \leq j \leq 3n$. Then, instead of the heads $\mathsf{HValid}_i(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y})$ in the mappings we can use

$$\exists u. \bigwedge_{j=1}^{3n} \mathsf{HValid}_i^j(u, v^j),$$

with v^j the j'th component of the vector $\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}$; the mappings with heads $\mathsf{VValid}_i(\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$ can be modified similarly.

To conclude the proof, we note that $\mathcal{D}_2 \models p_2$; thus, the notions of compliance and strong compliance coincide in this case. □

According to *Case (2)* of Theorem 19, the NEXPTIME upper bound for (strong) compliance can be improved to Π_3^p if the frontier of the mappings is bounded; if additionally the arity of the policy is bounded, then, according to *Case (3)* of the same theorem, the problems are in Σ_2^p. By *Cases (1)* and *(2)* of Theorem 21, the same bounds can be obtained if instead of bounded frontier we restrict the mappings to be LAV. In the next two theorems we give matching lower bounds for these algorithms. In the first of them, Theorem 25, we show that the problems are Π_3^p-hard even if the ontology is empty and the mappings are both CQ views and LAV with bounded source arity; if, additionally, the policy is Boolean, then the problems are Σ_2^p-hard. In the second one, Theorem 26, we show that if instead of linearity of mappings we restrict policies to be ground, the problem is also Σ_2^p-hard.

Theorem 25. *The problems* COMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *and* STRCOMPLY$(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *are*

(1) Π_3^p*-hard if* $\mathcal{O}$ *is empty,* $\mathcal{M}$ *ranges over a mapping language containing all sets of mappings that are CQ views, LAV and of source and global arities bounded by 2, and* p *ranges over a policy language containing all policies; and*

(2) Σ_2^p*-hard if the same restrictions as in Case (1) hold except that the source and global arities are bounded by 3, and the policy language contains all Boolean policies.*

Recall that in these settings the bound on the source arity implies the same bound on the frontier. As already mentioned, this theorem provides matching lower bounds for *Cases (2)* and *(3)* of Theorem 19 as well as *Cases (1)* and *(2)* of Theorem 21. By Theorems 21 and 22, these bounds cannot be improved along the dimensions considered in this paper; for example, if we additionally require the policy to be ground, then the problems are in AC^0 by *Case (b)* of Theorem 22.

Proof. First, for the setting of *Case (1)*, we show Σ_3^p-hardness of the complement of COMPLY by reduction of the $\exists\forall\exists$SAT problem. Then we argue that this reduction works essentially verbatim for showing Π_2^p-hardness of the complement of COMPLY for the setting of *Case (2)*. Note that, unlike other proofs in this section, these reductions do not directly work for STRCOMPLY, so in the end of the proof we show how to modify them for this problem.

We start with the Σ_3^p reduction. Let $\Phi = \exists\mathbf{s}.\forall\mathbf{u}.\exists\mathbf{v}.\Psi$ be a $\exists\forall\exists$3CNF formula, where $\mathbf{s}$, $\mathbf{u}$, and $\mathbf{v}$ are tuples of distinct propositional variables, while Ψ is a conjunction of clauses of the form $\ell_1 \vee \ell_2 \vee \ell_3$ for each ℓ_j either a variable from $\mathbf{s} \cup \mathbf{u} \cup \mathbf{v}$

or the negation of such a variable. We need to construct a set $\mathcal{M}$ of LAV CQ views of source and global arities bounded by 2, a source instance $\mathcal{D}$, and a CQ p such that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ if and only if Φ is $\texttt{true}$.

Similarly to the previous proofs, the source schema has a unary relational name Cl_γ for each clause γ in Ψ and binary relational names Arg_j for each $j = 1, 2, 3$, while source instance $\mathcal{D}$ uses a constant a^π_γ for each satisfying assignment π of variables of each clause γ in Ψ and constants f_w and t_w for each variable w in $\mathbf{s} \cup \mathbf{u} \cup \mathbf{v}$. Besides this, the source schema has binary relational names UVar_u and unary relational names Chosen_u for each $u \in \mathbf{u}$. The source instance also uses constants c^-_u and c^+_u for each $u \in \mathbf{u}$.

Next we define the source instance $\mathcal{D}$. Let it consist of the following facts:

- $\mathsf{Cl}_\gamma(a^\pi_\gamma)$, $\mathsf{Arg}_1(a^\pi_\gamma, b_1)$, $\mathsf{Arg}_2(a^\pi_\gamma, b_2)$, and $\mathsf{Arg}_3(a^\pi_\gamma, b_3)$, for each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$ in Ψ and satisfying assignment π of γ, where $b_j = f_w$ if $\pi(w) = \texttt{false}$ and $b_j = t_w$ if $\pi(w) = \texttt{true}$ with w the variable of ℓ_j,
- $\mathsf{UVar}_u(f_u, c^-_u)$, $\mathsf{UVar}_u(t_u, c^-_u)$, $\mathsf{UVar}_u(f_u, c^+_u)$, $\mathsf{UVar}_u(t_u, c^+_u)$, and $\mathsf{Chosen}_u(c^+_u)$, for each $u \in \mathbf{u}$.

Let policy p be the CQ over free variables x_s for $s \in \mathbf{s}$, and existential variables x_γ for γ in Ψ, x_w for $w \in \mathbf{u} \cup \mathbf{v}$ and y_u for $u \in \mathbf{u}$, with the following atoms:

$$\begin{array}{ll} \mathsf{Cl}_\gamma(x_\gamma), \mathsf{Arg}_j(x_\gamma, x_w), & \text{for each } \gamma = \ell_1 \vee \ell_2 \vee \ell_3 \text{ in } \Psi, \text{ each } j = 1, 2, 3, \text{ and each } w \in \mathbf{s} \cup \mathbf{u} \cup \mathbf{v} \text{ the variable of } \ell_j, \\ \mathsf{UVar}_u(x_u, y_u), \mathsf{Chosen}_u(y_u), & \text{for all } u \in \mathbf{u}. \end{array}$$

Next we define the global schema. It has "mirror" binary relational names Arg'_j for $j = 1, 2, 3$ and "mirror" unary relational names Cl'_γ for each γ in Ψ and Chosen'_u for each $u \in \mathbf{u}$. Besides these, the global schema contains unary relational names UVar'_u and Choice_u for each $u \in \mathbf{u}$.

Finally, let $\mathcal{M}$ consist of mappings

$$\begin{array}{rcll} \mathsf{Cl}_\gamma(x) & \to & \mathsf{Cl}'_\gamma(x), & \text{for all clauses } \gamma \text{ in } \Psi, \\ \mathsf{Arg}_j(x, y) & \to & \mathsf{Arg}'_j(x, y), & \text{for all } j = 1, 2, 3, \\ \mathsf{UVar}_u(x, y) & \to & \mathsf{UVar}'_u(x), & \text{for all } u \in \mathbf{u}, \\ \mathsf{UVar}_u(x, y) & \to & \mathsf{Choice}_u(y), & \text{for all } u \in \mathbf{u}, \\ \mathsf{Chosen}_u(x) & \to & \mathsf{Chosen}'_u(x), & \text{for all } u \in \mathbf{u}. \end{array}$$

Next we give an intuition for the reduction, assuming for simplicity that $\mathbf{s}$ is empty—that is, p is Boolean. Source instance $\mathcal{D}$ encodes all the satisfying assignments of all the clauses. In particular, it has a constant a^π_γ for each assignment π of variables of each clause γ, as well as constants f_w and t_w for $\texttt{true}$ and $\texttt{false}$ values of each propositional variable w (no matter whether it is universally or existentially quantified). Each universally quantified variable u is additionally associated with two constants c^-_u and c^+_u. Each constant a^π_γ is connected to the corresponding values of its variables by means of binary relational names Arg_j. Both constants f_u and t_u for the values of each universally quantified variable u are additionally connected to both of the corresponding constants c^-_u and c^+_u by relational name UVar_u. Mappings $\mathcal{M}$ copy all the source instance, except UVar_u, which is exported only by means of projections UVar'_u and Choice_u to the first and the second argument, respectively. Therefore, all the source instances indistinguishable from $\mathcal{D}$ differ from $\mathcal{D}$ only in UVar_u: the attacker knows that each of f_u and t_u, for all universally quantified u, is connected by UVar_u to at least one of c^-_u and c^+_u, and, conversely, each of c^-_u and c^+_u is connected to at least one of f_u and t_u. In other words, these indistinguishable source instances represent all possible assignments of universally quantified variables by means of the value constants connected to corresponding c^+_u (the source instances with both value constants of some u connected to c^+_u may be seen as representing several assignments). The task of the attacker is to check that the policy query p holds in all these source instances, which correspond to all possible assignments of $\mathbf{u}$. A homomorphism from p to any source instance sends each clause variable x_γ to one of its assignment variables a^π_γ, while each x_w to one of its value constants f_w and t_w. If w is existential then the choice is free. However, if w is universal, then x_w must be sent to the value constant that is connected to c^+_w, because only c^+_w is in Chosen_w as required by p. Therefore, such a homomorphism exists if and only if for all assignments to $\mathbf{u}$ (i.e., for all indistinguishable source instances) there exists an assignment to $\mathbf{v}$ that satisfies all the clauses of Ψ.

Next, we formally show the correctness of our reduction. For the forward direction, suppose first that Φ holds—that is, there exists an assignment of $\mathbf{s}$ such that for any assignment of $\mathbf{u}$ there is an assignment of $\mathbf{v}$ such that Ψ evaluates to $\texttt{true}$. We need to show that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p)$ does not hold—that is, there exists a tuple of constants $\mathbf{a}$ of the same size as $\mathbf{s}$ such that for each source instance $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ there is a homomorphism from $p(\mathbf{a})$ to $\mathcal{D}'$. We first construct $\mathbf{a}$, and do it according to the assignment σ of $\mathbf{s}$ in the straightforward way: if $\sigma(s) = \texttt{false}$ for some s in $\mathbf{s}$ then the corresponding a in $\mathbf{a}$ is f_s, and if $\sigma(s) = \texttt{true}$ then a is t_s. Consider now any $\mathcal{D}'$ indistinguishable from $\mathcal{D}$. By construction, for each $u \in \mathbf{u}$ at least one of $\mathsf{UVar}_u(f_u, c^+_u)$ and $\mathsf{UVar}_u(t_u, c^+_u)$ is in $\mathcal{D}'$. Extend the assignment σ to $\mathbf{u}$ in such a way that, for every u, $\sigma(u) = \texttt{false}$ if $\mathsf{UVar}_u(f_u, c^+_u)$ is in $\mathcal{D}'$ and $\sigma(u) = \texttt{true}$ otherwise. Since Φ holds, σ can be extended to $\mathbf{v}$ such that Ψ evaluates to $\texttt{true}$ under the overall σ. Taking such an extension, let, for every $w \in \mathbf{u} \cup \mathbf{v}$, d_w be f_w if $\sigma(w) = \texttt{false}$ and t_w otherwise. Consider now the mapping h of variables of $p(\mathbf{a})$ to constants of $\mathcal{D}'$ such that

1. $h(x_\gamma) = a^\pi_\gamma$, for each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$, where π is the assignment mapping w of each ℓ_j to $\sigma(w)$;

2. $h(x_w) = d_w$, for each $w \in \mathbf{u} \cup \mathbf{v}$; and

3. $h(y_u) = c_u^+$, for each $u \in \mathbf{u}$.

By construction, h is a homomorphism from $p(\mathbf{a})$ to $\mathcal{D}'$.

Suppose now that $\text{COMPLY}(\emptyset, \mathcal{M}, \mathcal{D}, p)$ does not hold—that is, there exists a tuple $\mathbf{a}$ of constants of the same size as $\mathbf{s}$ such that for each source instance $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ there is a homomorphism from $p(\mathbf{a})$ to $\mathcal{D}'$. We need to show that Φ is `true`—that is, there exists an assignment of $\mathbf{s}$ such that for any assignment of $\mathbf{u}$ there exists an assignment of $\mathbf{v}$ such that Ψ evaluates to `true` under the overall assignment. By construction, the existence of a homomorphism implies that each constant a in $\mathbf{a}$ is either f_s or t_s for the corresponding variable s in $\mathbf{s}$. Let σ be the assignment of $\mathbf{s}$ such that $\sigma(s) = \texttt{false}$ if the corresponding a is f_s, and $\sigma(s) = \texttt{true}$ otherwise. Consider any extension of the assignment σ to variables $\mathbf{u}$. By construction, there is an indistinguishable $\mathcal{D}'$ such that, for any $u \in \mathbf{u}$, $\mathsf{UVar}_u(f_u, c_u^+)$ is in $\mathcal{D}'$ and $\mathsf{UVar}_u(t_u, c_u^+)$ is not if $\sigma(u) = \texttt{false}$ and vice versa otherwise. Consider a homomorphism h from $p(\mathbf{a})$ to $\mathcal{D}'$. Again by construction, it agrees with σ in the sense that, for every u, $h(x_u) = f_u$ if $\sigma(u) = \texttt{false}$ and $h(x_u) = t_u$ otherwise. Extend σ to $\mathbf{v}$ in the same way: let, for each $v \in \mathbf{v}$, $\sigma(v) = \texttt{false}$ if $h(x_v) = f_v$ and $\sigma(v) = \texttt{true}$ otherwise. By construction, σ is an assignment satisfying Ψ, as required.

For *Case (2)*, note that we can reuse the reduction for *Case (1)* when $\mathbf{s}$ is empty and p is Boolean. In fact, the intuition for the construction above is given for this simplified setting.

As already mentioned, unlike other proofs in this section, these reductions do not apply as is to STRCOMPLY. The problem is that, for some tuples of constants $\mathbf{a}$, $p(\mathbf{a})$ does not hold in the constructed $\mathcal{D}$. This may happen even in *Case (2)*—that is, when p is Boolean—because we cannot guarantee that Ψ is satisfiable. It is harmless for COMPLY, because we just look for an indistinguishable source instance in which $p(\mathbf{a})$ holds, regardless of the fact whether it holds in $\mathcal{D}$ or not. But for STRCOMPLY, in case of $\mathcal{D} \not\models p(\mathbf{a})$, we need to find an indistinguishable instance in which $p(\mathbf{a})$ does hold, which may be problematic for the reduction: any attempt to satisfy $p(\mathbf{a})$ in a source instance may make it distinguishable due to fact that $\mathcal{M}$ exports everything for almost all the relational names in p.

Therefore, we modify the reduction as follows. First, we increase the arity of all relational names in the source schema by one—that is, unary names, such as Cl_γ, become binary and binary names, such as Arg_j, become ternary (however, the target relational names stay intact). Second, the source instance $\mathcal{D}$ just repeats the first argument twice in each fact, for example, $\mathsf{UVar}_u(f_u, c_u^-)$ becomes $\mathsf{UVar}_u(f_u, f_u, c_u^-)$. Third, the mappings $\mathcal{M}$ are modified in the similar way, by repeating the first variable in the body atom twice; for example, we obtain the mapping $\mathsf{Arg}_j(x, x, y) \to \mathsf{Arg}'_j(x, y)$. Finally, policy p does not repeat the variables; instead, it uses a fresh existentially quantified variable as the second argument of each atom; for example, it has an atom $\mathsf{UVar}_u(x_u, z, y_u)$ instead of $\mathsf{UVar}_u(x_u, y_u)$, where z does not appear anywhere else in the body of p.

The resulting reduction still works for COMPLY, with the same justification. However, now we can easily find an indistinguishable instance in which $p(\mathbf{a})$ holds for any tuple $\mathbf{a}$ such that $\mathcal{D} \not\models p(\mathbf{a})$. For example, $\mathcal{D} \cup \mathcal{D}_{p(\mathbf{a})}$ would serve the need, where $\mathcal{D}_{p(\mathbf{a})}$ is the set of all facts obtained from atoms of $p(\mathbf{a})$ be replacing each existential variable by a fresh constant. Indeed, $\mathcal{D} \cup \mathcal{D}_{p(\mathbf{a})}$ is indistinguishable from $\mathcal{D}$ because the facts in $\mathcal{D}_{p(\mathbf{a})}$ are not exported by $\mathcal{M}$ since they have different first two arguments. Moreover, $\mathcal{D} \cup \mathcal{D}_{p(\mathbf{a})} \models p(\mathbf{a})$ by construction. □

We now focus on another Σ_2^p lower bound for (strong) compliance, which is incomparable to the one of *Case (2)* of Theorem 25: instead of requiring mappings to be LAV it requires the policy to be a fact (i.e., ground).

Theorem 26. *The problems* $\text{COMPLY}(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *and* $\text{STRCOMPLY}(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *are* Σ_2^p*-hard if* $\mathcal{O}$ *is empty,* $\mathcal{M}$ *ranges over a mapping language containing all sets of CQ views of source and global arities bounded by 2, and* p *ranges over a policy language containing all facts.*

Recall again that the bound on the global arity in this setting implies the same bound on the frontier. As already mentioned, this theorem provides a matching lower bound for *Case (3)* of Theorem 19 and *Case (2)* of Theorem 21; this bound is incomparable with the bound in *Case (2)* of Theorem 25. By Theorems 21 and 22, these bounds cannot be improved along the dimensions considered in this paper; for example, if we join the requirements for these two Σ_2^p lower bounds, then the problems are in AC^0 by *Case (b)* of Theorem 22.

Proof. We show Π_2^p-hardness of the complement of COMPLY by reduction of the $\forall\exists$3SAT problem. Our proof applies verbatim to STRCOMPLY since our reduction always constructs an instance of compliance where $\mathcal{D} \models p$.

Let $\Phi = \forall\mathbf{u}.\, \exists\mathbf{v}.\, \Psi$, where $\mathbf{u}$ and $\mathbf{v}$ are tuples of distinct propositional variables, and Ψ is a conjunction of clauses of the form $\ell_1 \vee \ell_2 \vee \ell_3$ for ℓ_j either a variable from $\mathbf{u} \cup \mathbf{v}$ or the negation of such a variable. We need to construct a set of CQ views $\mathcal{M}$ of source and global arities bounded by 2, a source instance $\mathcal{D}$, and a fact p such that $\text{COMPLY}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ if and only if Φ is `true`.

Similarly to the Π_2^p-hardness proof of Theorem 14, the source schema has a unary relational name Cl_γ for each clause γ in Ψ as well as binary relational names Arg_j for $j = 1, 2, 3$, while source instance $\mathcal{D}$ uses a constant a_γ^π for each satisfying assignment π of variables of each clause γ in Ψ and constants f_w and t_w for each variable w in $\mathbf{u} \cup \mathbf{v}$ (recall that there are at most 7 satisfying

assignments for each clause). Besides this, the source schema has unary relational names Choice, $\mathsf{Trigger}$, Check, as well as Value_u for each $u \in \mathbf{u}$ and $\mathsf{Assigned}_w$ for each $w \in \mathbf{u} \cup \mathbf{v}$. The source instance also uses two constants d and e.

Next we define the source instance $\mathcal{D}$. Let it consist of the following facts:

- $\mathsf{Cl}_\gamma(a^\pi_\gamma)$, $\mathsf{Arg}_1(a^\pi_\gamma, b_1)$, $\mathsf{Arg}_2(a^\pi_\gamma, b_2)$, and $\mathsf{Arg}_3(a^\pi_\gamma, b_3)$, for each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$ in Ψ and satisfying assignment π of γ, where $b_j = f_w$ if $\pi(w) = \texttt{false}$ and $b_j = t_w$ if $\pi(w) = \texttt{true}$ with w the variable of ℓ_j,
- $\mathsf{Value}_u(f_u)$ and $\mathsf{Value}_u(t_u)$, for each $u \in \mathbf{u}$,
- $\mathsf{Assigned}_w(f_w)$ and $\mathsf{Assigned}_w(t_w)$, for each $w \in \mathbf{u} \cup \mathbf{v}$,
- $\mathsf{Choice}(d)$, $\mathsf{Trigger}(d)$, $\mathsf{Choice}(e)$, and $\mathsf{Check}(e)$.

Let also the policy p be the fact $\mathsf{Trigger}(d)$.

Next we define the global schema. It has "mirror" binary relational names Arg'_j for $j = 1, 2, 3$, and "mirror" unary relational names Choice', Check', as well as Cl'_γ for each γ in Ψ, Value'_u for each $u \in \mathbf{u}$ and $\mathsf{Assigned}'_v$ for each $v \in \mathbf{v}$. In addition to these, the global schema contains nullary $\mathsf{Trigger}'$, $\mathsf{HasAssignment}$, and $\mathsf{Assigned}'_u$ for each $u \in \mathbf{u}$.

Let $\mathcal{M}$ consist of the following mappings:

$$\begin{aligned}
\mathsf{Cl}_\gamma(x) &\to \mathsf{Cl}'_\gamma(x), && \text{for all clauses } \gamma \text{ in } \Psi,\\
\mathsf{Arg}_j(x,y) &\to \mathsf{Arg}'_j(x,y), && \text{for all } j = 1,2,3,\\
\mathsf{Value}_u(x) &\to \mathsf{Value}'_u(x), && \text{for all } u \in \mathbf{u},\\
\mathsf{Assigned}_u(x) \wedge \mathsf{Value}_u(x) &\to \mathsf{Assigned}'_u(), && \text{for all } u \in \mathbf{u},\\
\mathsf{Assigned}_v(x) &\to \mathsf{Assigned}'_v(x), && \text{for all } v \in \mathbf{v},\\
\mathsf{Choice}(x) &\to \mathsf{Choice}'(x),\\
\mathsf{Trigger}(x) \wedge \mathsf{Choice}(x) &\to \mathsf{Trigger}'(),\\
\mathsf{Check}(x) &\to \mathsf{Check}'(x),
\end{aligned}$$

and the mapping

$$\mathsf{Check}(x) \wedge \mathsf{Trigger}(x) \wedge \xi \to \mathsf{HasAssignment}(),$$

where ξ is the conjunction of the following atoms, for each clause $\gamma = \ell_1 \vee \ell_2 \vee \ell_3$ in Ψ:

$$\mathsf{Cl}_\gamma(x_\gamma),\ \mathsf{Arg}_j(x_\gamma, x_w), \mathsf{Assigned}_w(x_w), \quad \text{for all } 1 \le j \le 3 \text{ and for the variable } w \in \mathbf{u} \cup \mathbf{v} \text{ of } \ell_j.$$

We now show correctness of the construction, namely that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ if and only if Φ is $\texttt{true}$. First observe that, by knowing the virtual image of $\mathcal{D}$ and $\mathcal{M}$, attackers know all of $\mathcal{D}$ except atoms over $\mathsf{Assigned}_u$ and $\mathsf{Trigger}$; instead, they know that at least one of $\mathsf{Trigger}(d)$ and $\mathsf{Trigger}(e)$, and, for each $u \in \mathbf{u}$, at least one of $\mathsf{Assigned}_u(f_u)$ and $\mathsf{Assigned}_u(t_u)$ hold. Furthermore, if $\mathsf{Trigger}(e)$ holds in some source instance $\mathcal{D}'$, then the shape of the last mapping in $\mathcal{M}$ ensures that $\mathsf{HasAssignment}()$ is exported only if $\mathcal{D}'$ contains either $\mathsf{Assigned}_u(f_u)$ or $\mathsf{Assigned}_u(t_u)$ for every $u \in \mathbf{u}$. Since $\mathsf{HasAssignment}()$ is not in the image of $\mathcal{D}$, the shape of ξ also ensures that the following properties hold:

- if Φ is $\texttt{true}$ and a source instance $\mathcal{D}'$ without $\mathsf{Trigger}(d)$ is indistinguishable from $\mathcal{D}$, then $\mathcal{D}' \models \xi$,
- if Φ is $\texttt{false}$ then there exists an indistinguishable $\mathcal{D}'$ without $\mathsf{Trigger}(d)$ such that $\mathcal{D}' \not\models \xi$.

With this in mind, let us assume that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$. This implies that every $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ satisfies $\mathsf{Trigger}(d)$—that is, the policy. Let us assume for the sake of contradiction that Φ is $\texttt{false}$; by the properties above we have that there exists a source instance $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ such that $\mathcal{D}' \not\models \xi$; but then, we additionally have the freedom to require that $\mathsf{Trigger}(e) \in \mathcal{D}'$ but $\mathsf{Trigger}(d) \notin \mathcal{D}'$ without breaking indistinguishability. Such a $\mathcal{D}'$ is hence a witness to compliance, which yields a contradiction to our assumption that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$.

For the converse, assume that Φ is $\texttt{true}$. By the properties above we have that $\mathcal{D}' \models \xi$ for every $\mathcal{D}'$ indistinguishable from $\mathcal{D}$. But then, we must require, on the one hand, that $\mathsf{Trigger}(e) \notin \mathcal{D}'$ (otherwise the image of $\mathcal{D}'$ would contain $\mathsf{HasAssignment}()$ thus breaking indistinguishability) and, on the other hand, that $\mathsf{Trigger}(d) \in \mathcal{D}'$ (in order to ensure that the image of $\mathcal{D}'$ contains $\mathsf{Trigger}'()$). As a result, every $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ must include the policy and hence $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$, as required. □

Case (3) of Theorem 21 gives us an NP upper bound for compliance and strong compliance when the mappings are LAV and the policy is ground. In Proposition 23 we have already obtained a matching lower bound, even in the case when additionally the ontology is empty and the source and global arities (and, hence, the frontier) are bounded. We next prove an incomparable

matching lower bound, for the case when instead of the bound on the source arity we restrict the mappings to be sets of CQ views (note that the global arity and hence the frontier can still be bounded).

We make use of the following variation of the satisfiability problem: let NONTOTALSAT take as input a propositional formula Ψ in CNF such that the assignment of all its variables to `true` is satisfying, and returns `true` if and only if there is a satisfying assignment of Ψ mapping at least one variable to `false`.

Lemma 27. *The problem* NONTOTALSAT(Ψ) *is* NP*-complete.*

Proof. Membership in NP is immediate, and hardness is justified by a simple reduction of the standard SAT problem. Specifically, given a propositional formula Φ in CNF, the reduction distinguishes the following two cases. First, if the assignment of all variables of Φ to `true` is satisfying, then $\Psi = u_1 \vee u_2$ for fresh propositional variables u_1 and u_2. Otherwise, Ψ is the CNF of the formula $(v_1 \wedge \cdots \wedge v_n) \vee \Phi$, where $v_1, \ldots, v_n$ are all the variables of Φ. Note that conversion to CNF increases the size of the formula at most quadratically. It is immediate to check that SAT(Φ) = `true` if and only if NONTOTALSAT(Ψ) = `true`. □

Theorem 28. *The problems* COMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *and* STRCOMPLY($\mathcal{O}, \mathcal{M}, \mathcal{D}, p$) *are* NP*-hard if $\mathcal{O}$ is empty, $\mathcal{M}$ ranges over a mapping language containing all sets of mappings that are CQ views, LAV and of global arity bounded by 2, and p ranges over a policy language containing all facts.*

Recall that in this setting the bound on the global arity implies the same bound on the frontier. This theorem provides a matching lower bound for *Case (3)* of Theorem 21, which is incomparable to the bound from *Case (a)* of Proposition 23. By Theorem 22, these bounds cannot be improved along the dimensions considered in this paper; for example, if we join the requirements for the two lower bounds, then the problems are in AC^0 by *Case (b)* of Theorem 22.

Proof. The proof for COMPLY is by reduction of the NONTOTALSAT problem, which is NP-hard by Lemma 27. As usual, the source instance in the reduction satisfies the policy, so the same reduction works also for STRCOMPLY. Let Ψ be a propositional formula in CNF over variables $v_1, \ldots, v_n$ such that the assignment of all its variables to `true` is satisfying.

The source schema in the reduction consists of a single relational name $\mathsf{Assignment}$ of arity $n + 4$. Then, let $\mathcal{D}$ consist of the following facts over constants 0 and 1:

$$\begin{aligned} \alpha &= \mathsf{Assignment}(1, 1, 0, 1, 1, \ldots, 1), \\ \beta &= \mathsf{Assignment}(0, 1, 0, 1, 0, \ldots, 0). \end{aligned}$$

Also, let policy p be fact α.

The global schema consists of a binary relational name Header, unary relational names Zero, One, and Value_i for each $i = 1, \ldots, n$, and nullary relational names $\mathsf{Falsified}_\gamma$ for each clause γ in Ψ. Then, let $\mathcal{M}$ contain mappings

$$\begin{aligned} \mathsf{Assignment}(z_1, z_2, y_{zero}, y_{one}, x_{v_1}, \ldots, x_{v_n}) &\rightarrow \mathsf{Header}(z_1, z_2), \\ \mathsf{Assignment}(z_1, z_2, y_{zero}, y_{one}, x_{v_1}, \ldots, x_{v_n}) &\rightarrow \mathsf{Zero}(y_{zero}), \\ \mathsf{Assignment}(z_1, z_2, y_{zero}, y_{one}, x_{v_1}, \ldots, x_{v_n}) &\rightarrow \mathsf{One}(y_{one}), \end{aligned}$$

the mappings

$$\mathsf{Assignment}(z_1, z_2, y_{zero}, y_{one}, x_{v_1}, \ldots, x_{v_n}) \rightarrow \mathsf{Value}_i(x_{v_i}), \qquad \text{for each } 1 \leq i \leq n,$$

and, assuming without loss of generality that the bodies of mappings can have equalities of variables as conjuncts, the mapping

$$\mathsf{Assignment}(z_1, z_2, y_{zero}, y_{one}, x_{v_1}, \ldots, x_{v_n}) \wedge (z_1 = z_2) \wedge (x_{v^1} = t_1) \wedge \cdots \wedge (x_{v^k} = t_k) \rightarrow \mathsf{Falsified}_\gamma(),$$

for each clause γ in Ψ over variables $v^1, \ldots, v^k$ among $v_1, \ldots, v_n$, where, for each $i = 1, \ldots, k$, $t_i = y_{zero}$ if $\pi(v_i^\gamma)$ = `false` and $t_i = y_{one}$ if $\pi(v_i^\gamma)$ = `true` for the only assignment π of $v^1, \ldots, v^k$ falsifying γ.

Intuitively, the third position of $\mathsf{Assignment}$ stores the value 0 and the fourth stores the value 1 (these positions are included in the relation just to avoid the use of constants in the mappings). A tuple in $\mathsf{Assignment}$ in which the first two "header" elements are equal represents an assignment, given by the final n positions. Call such an assignment $\mathsf{Assignment}$*-generated*. In $\mathcal{D}$ the only $\mathsf{Assignment}$-generated assignment sends all variables to `true`, which is assumed to be a satisfying. The last set of mappings then guarantee that in an indistinguishable instance all $\mathsf{Assignment}$-generated assignments must be satisfying. The policy guarantees that we are interested in indistinguishable instances where we do not have an $\mathsf{Assignment}$-generated assignment that has all variables set to `true`.

We claim that $\mathsf{NonTotalSAT}(\Psi)$ is `true` if and only if COMPLY($\emptyset, \mathcal{M}, \mathcal{D}, p$) is `true`. Indeed, consider first the virtual image $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ available to the attacker:

$$\begin{aligned} &\mathsf{Header}(1, 1), \mathsf{Header}(0, 1), \\ &\mathsf{Zero}(0), \mathsf{One}(1), \\ &\mathsf{Value}_i(0), \mathsf{Value}_i(1), \qquad \text{for each } 1 \leq i \leq n. \end{aligned}$$

In particular, $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ does not contain any of $\mathsf{Falsified}_\gamma()$ facts, since, by assumption, fact α represents a satisfying assignment, while fact β has distinct values at first two positions and, hence, does not trigger any of the corresponding mappings.

In other words, the attacker knows that

1. there are at least two $\mathsf{Assignment}$ facts in the source, one with $1, 1$ at the first two positions, and another with $0, 1$;
2. all facts have 0 at the third position and 1 at the fourth;
3. all facts have either 0 or 1 at every position $i = 5, \ldots, n + 4$; moreover, for each position and each constant 0 and 1 there is a fact with with the constant at the position; and
4. the facts with $1, 1$ at the first two positions encode satisfying assignments by values at positions $5, \ldots, n + 4$.

Thus, if $\textsc{NonTotalSat}(\Psi) = \texttt{true}$, then an indistinguishable source instance contains facts

$$\mathsf{Assignment}(1, 1, 0, 1, a_1, \ldots, a_n),$$
$$\mathsf{Assignment}(0, 1, 0, 1, b_1, \ldots, b_n),$$

where $a_1, \ldots, a_n$ encode the satisfying assignment different from the one disclosing the policy, and each b_i is the complement of a_i (note that the assignment encoded by $b_1, \ldots, b_n$ need not to be satisfying, because the falsifying mappings require the first two parameters of $\mathsf{Assignment}$ to be equal). Conversely, if $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{true}$ then a second satisfying assignment of Ψ is encoded in a fact with $1, 1$ as the first two parameters in the indistinguishable source instance not containing the policy fact. □

Theorems 24, 25, 26, and 28, together with Proposition 23, complete the picture for combined complexity of compliance and strong compliance along all the dimensions considered in this paper. In what follows we do the same for data complexity, where the ontology $\mathcal{O}$, mappings $\mathcal{M}$ and policy p are considered fixed and only the source instance $\mathcal{D}$ forms the input.

From Theorems 19 and 22, we know that both problems inherit the upper bound for data complexity of CQ (or fact) entailment for the ontology language if this complexity includes NP or if the mappings are LAV (or both GAV and LAV, respectively) while the policy is ground. In particular, for the empty ontology, the problems are in NP in general and in AC^0 if the mappings are LAV while the policy is ground. Next we prove two more incomparable matching lower bounds for the general NP upper bound of *Case (4)* of Theorem 19, one when the mappings are CQ views and LAV, but the policy is arbitrary, and one when the mappings are arbitrary CQ views, but the policy is ground.

Theorem 29. *In data complexity, the problems* $\textsc{Comply}(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *and* $\textsc{StrComply}(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ *are* NP*-hard if* $\mathcal{O}$ *is empty,* $\mathcal{M}$ *ranges over a mapping language* $\mathbb{M}$*, and* p *ranges over a policy language* $\mathbb{P}$ *such that either*

(a) $\mathbb{M}$ *contains all sets of CQ views and* $\mathbb{P}$ *contains all facts, or*

(b) $\mathbb{M}$ *contains all sets of mappings that are CQ views and LAV, and* $\mathbb{P}$ *contains all Boolean policies.*

Note that source arity, global arity, and frontier are implicit bounded when studying data complexity. As already mentioned, this theorem provides a matching lower bound for *Case (4)* of Theorem 19.

Proof. In both cases, the policy constructed in the reduction holds in the source instance, so the proof works for both compliance and strong compliance. As usual, we concentrate on the first.

In *Case (a)* we reduce NonTotalSat problem, which is NP-complete by Lemma 27. In particular, we give a set of CQ views $\mathcal{M}$ and a fact α_p such that for any propositional formula Ψ in CNF such that the assignment of all its variables to $\texttt{true}$ is satisfying we can construct, in polynomial time, a source instance $\mathcal{D}$ such that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, \alpha_p)$ is $\texttt{true}$ if and only if $\textsc{NonTotalSat}(\Psi)$ is $\texttt{true}$.

Let the source schema consist of binary relational names Arg, Value, and $\mathsf{Assigned}$, as well as unary relational names Zero, One, and SZero. Then, let the global schema consist of "mirror" binary relational names Arg' and Value', unary relational names Zero', One', $\mathsf{HasAssigned}$, Valid, SZero' and SOne', and nullary relational names Clash and Secret.

We now construct the CQ views and the policy. In particular, let $\mathcal{M}$ be the (non-LAV) CQ view mappings

$$\begin{aligned}
\mathsf{Arg}(x, y) &\rightarrow \mathsf{Arg}'(x, y),\\
\mathsf{Zero}(y) &\rightarrow \mathsf{Zero}'(y),\\
\mathsf{One}(y) &\rightarrow \mathsf{One}'(y),\\
\mathsf{Value}(z, y) &\rightarrow \mathsf{Value}'(z, y),\\
\mathsf{Value}(z, y) \wedge \mathsf{Assigned}(z, y) &\rightarrow \mathsf{HasAssigned}(z),\\
\mathsf{Assigned}(z, y) \wedge \mathsf{Zero}(y) \wedge \mathsf{Assigned}(z, y') \wedge \mathsf{One}(y') &\rightarrow \mathsf{Clash}(),\\
\mathsf{Arg}(x, y) \wedge \mathsf{Value}(z, y) \wedge \mathsf{Assigned}(z, y) &\rightarrow \mathsf{Valid}(x),\\
\mathsf{SZero}(y) \wedge \mathsf{Zero}(y) &\rightarrow \mathsf{SZero}'(y),\\
\mathsf{SZero}(y) \wedge \mathsf{One}(y) &\rightarrow \mathsf{SOne}'(y),\\
\mathsf{Value}(z, y) \wedge \mathsf{Assigned}(z, y) \wedge \mathsf{SZero}(y) &\rightarrow \mathsf{Secret}(),
\end{aligned}$$

and let α_p be the fact $\mathsf{SZero}(s)$, for a constant s.

Consider now a propositional formula Ψ in CNF over variables $\mathbf{v}$ such that the assignment of all its variables to `true` is satisfying. Based on this, let $\mathcal{D}$ consist of the following atoms, over constants a_γ for each clause γ in Ψ, f_v and t_v for each $v \in \mathbf{v}$, as well as a and s (where s is already used in the policy):

$$\begin{aligned}
\mathsf{Arg}(a_\gamma, f_v), &\quad \text{for each clause } \gamma \text{ in } \varphi \text{ and each negative literal } \neg v \text{ in } \gamma,\\
\mathsf{Arg}(a_\gamma, t_v), &\quad \text{for each clause } \gamma \text{ in } \varphi \text{ and each positive literal } v \text{ in } \gamma,\\
\mathsf{Zero}(f_v), \mathsf{One}(t_v), &\quad \text{for each } v \in \mathbf{v},\\
\mathsf{Value}(a_v, f_v), \mathsf{Value}(a_v, t_v), &\quad \text{for each } v \in \mathbf{v},\\
\mathsf{Assigned}(a_v, t_v), \mathsf{SZero}(f_v), &\quad \text{for each } v \in \mathbf{v},\\
\mathsf{Value}(a, s), \mathsf{Assigned}(a, s), \mathsf{SZero}(s).&
\end{aligned}$$

We claim that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, \alpha_p)$ is `true` if and only if $\textsc{NonTotalSat}(\Psi)$ is `true`. Informally, the conjunction of Assigned and Value represents a desired truth assignment. Consider the virtual image $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ available to the attacker. A major difference from $\mathcal{D}$ is that Assigned is projected to the first argument and SZero is exported for all f_v, but not for s (plus, $\mathsf{Valid}(a_\gamma)$ for all γ and Secret are exported). Therefore, the attacker knows that

- all variables are assigned to `false` or `true` (by means of the mapping with HasAssigned head),
- no variables are assigned to both, because Clash( ) is not exported (note that, in $\mathcal{D}$, Value associates both `true` and `false` to a variable, but Assigned only `true`),
- the assignment is satisfying, because all $\mathsf{Valid}(a_\gamma)$ are exported,
- either one of the variables are assigned to `false`, or the policy holds, because Secret( ) is exported.

In other words, either there exists a satisfying assignment with a `false` variable, or the policy is disclosed, as required.

Formally, assume first that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{true}$. We need to show that $\textsc{NonTotalSat}(\Psi) = \texttt{true}$. Let $\mathcal{D}'$ be an indistinguishable source instance without the policy fact $\mathsf{SZero}(s)$. By the observation above, the constant corresponding to every propositional variable is related by Assigned and Value to either a constant satisfying Zero or a constant satisfying One, but not both. Thus we can define a truth assignment to variables accordingly. Also by the observation, this assignment is satisfying. We now claim that there exists v such that $\mathsf{Assigned}(a_v, f_v)$ holds—that is, the assignment has a variable assigned to `false`. Indeed, Secret( ) is exported, so $\mathsf{Value}(z, y) \wedge \mathsf{Assigned}(z, y) \wedge \mathsf{SZero}(y)$ should hold for some z and y. Since there is a mapping exporting Value by its own, (z, y) can be only either (a, s) or one of (a_v, f_v) and (a_v, t_v) for some $v \in \mathbf{v}$. The first is not possible because policy $\mathsf{SZero}(s)$ does not hold in $\mathcal{D}'$. None of (a_v, t_v) is possible, because otherwise we would have $\mathsf{SOne}'(y)$ exported. So, the only possibility is that $\mathsf{Value}(a_v, f_v) \wedge \mathsf{Assigned}(a_v, f_v) \wedge \mathsf{SZero}(f_v)$ holds for some v. Also, $\mathsf{Assigned}(a_v, t_v)$ cannot hold for this v, because otherwise we would export Clash( ). Thus, v is assigned to `false` in the assignment, as required.

For the converse direction, let $\textsc{NonTotalSat}(\Psi) = \texttt{true}$. We need to show that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, \alpha_p) = \texttt{true}$. Consider an assignment σ to $\mathbf{v}$ satisfying Ψ with at least one v in $\mathbf{v}$ such that $\sigma(v) = \texttt{false}$. Let $\mathcal{D}'$ be the source instance that agrees with $\mathcal{D}$ on Arg, Zero, One and Value, as well as contains the following atoms:

$$\begin{aligned}
\mathsf{Assigned}(a_v, f_v), &\quad \text{for each } v \in \mathbf{v} \text{ with } \sigma(v) = \texttt{false},\\
\mathsf{Assigned}(a_v, t_v), &\quad \text{for each } v \in \mathbf{v} \text{ with } \sigma(v) = \texttt{true},\\
\mathsf{SZero}(f_v), &\quad \text{for each } v \in \mathbf{v},\\
\mathsf{Assigned}(a, s).&
\end{aligned}$$

It is immediate to check that source instance $\mathcal{D}'$ is indistinguishable and does not contain the policy fact, as required.

In *Case (b)*, we reduce the 3-Colourability problem, whose input is a (directed) graph G and $\textsc{3-Colourability}(G)$ is `true` if and only if G is 3-colourable. In particular, we give a set of mappings $\mathcal{M}$ that are both CQ views and LAV, and a Boolean policy p such that for any directed graph G we can construct, in polynomial time, a source instance $\mathcal{D}$ such that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p)$ is `true` if and only if $\textsc{3-Colourability}(G)$ is `true`. The reduction is analogous to that in Lemma 3.8 in [54] for the problem of instance-based determinacy (see Section 8 for a discussion on the connection between this problem and policy compliance).

Let the source schema consist of two binary relational names Edge and Colour, and the global schema consists of a "mirror" binary relational name Edge′ and "projection" unary relational names Node and Colour′. Let then the set of CQ views $\mathcal{M}$ has mappings

$$\mathsf{Edge}(x, y) \to \mathsf{Edge}'(x, y), \qquad \mathsf{Colour}(x, y) \to \mathsf{Node}(x), \qquad \mathsf{Colour}(x, y) \to \mathsf{Colour}'(y),$$

and the policy p be the Boolean CQ

$$\mathsf{Edge}(x, y) \wedge \mathsf{Colour}(x, z) \wedge \mathsf{Colour}(y, z).$$

Consider an instance of 3-colourability consisting of a graph $G = (V, E)$ with vertices V and edges E. Without loss of generality we assume that it has at least one edge. Then, the source instance $\mathcal{D}$ uses constants r, g, b, and a_v for each $v \in V$, and consists of facts

$$\begin{array}{ll} \mathsf{Edge}(a_{v_1}, a_{v_2}), & \text{for each } (v_1, v_2) \in E, \text{ and} \\ \mathsf{Colour}(a_v, c), & \text{for each } v \in V \text{ and each } c \in \{r, g, b\}. \end{array}$$

It is immediate to see that G is 3-colourable if and only if $\textsc{Comply}(\mathcal{O}, \mathcal{M}, \mathcal{D}, p) = \texttt{true}$. □

7. Data-Independent Compliance

We now turn our attention to our data-independent notions of compliance defined in Section 3.4, which require that all possible source instances (strongly) comply to the policy. This is a very desirable property for (the schema of) a data integration system to satisfy: it ensures that the policy is not revealed to a malicious attacker, regardless of the underlying source instance (and hence regardless of any updates in the source instances). Recall that in the data-independent case we consider only ComplyAll leaving StrComplyAll for future work. We also concentrate on constant-free ontologies and mappings, as well as policies that are Boolean CQs and also constant-free.

Unfortunately, we show in Section 7.1 that the data-independent compliance problem is undecidable under rather strong restrictions on the input. Then, in Section 7.2 we identify a decidable case and also a tractable restriction thereof.

7.1. Undecidability

In this section, we show that ComplyAll is undecidable, even for empty ontology, GAV mappings, and bounded arity of the global schema. As an immediate corollary, we obtain that the data-independent compliance problems are undecidable even for CQ views in the presence of ontologies that can enforce equivalence of global relations. Note that full TGDs, linear TGDs, and DL-Lite$_R$ satisfy this condition.

The proof is via an involved reduction of the well-known domino tiling problem [9], which we elaborate in the remainder of this section. Our reduction exploits a variant of the "challenge method" by Benedikt et al. [7], where special "challenge" relational names are introduced in the mappings and policy to ensure confluence and hence close the grid.

Theorem 30. *Problem* $\textsc{ComplyAll}(\mathcal{O}, \mathcal{M}, p)$ *is undecidable if* $\mathcal{O}$ *is empty,* $\mathcal{M}$ *ranges over a mapping language containing all mappings that are contant-free, GAV and of source and global arities bounded by 2, and* p *ranges over a policy language containing all constant-free and Boolean policies.*

Recall that in this setting the bound on the global arity implies the same bound on the frontier.

Proof. First, we give a reduction of the undecidable problem of tiling an infinite grid to the complement of data-independent compliance problem assuming that p may be a Boolean UCQ and source instances are allowed to be infinite; later on, we discuss how to adapt the proof to dispense with the additional assumptions.

Similarly to the NExpTime-complete version of the tiling problem used in the proof of Theorem 24, the undecidable version takes as input a tiling instance $(\mathcal{T}, R_H, R_V)$, with a finite set of tile types $\mathcal{T}$, horizontal compatibility relation $R_H \subseteq \mathcal{T} \times \mathcal{T}$ and vertical compatibility relation $R_V \subseteq \mathcal{T} \times \mathcal{T}$. The instance *has a solution* if and only if it is possible to tile the whole positive quarter $\mathbb{N} \times \mathbb{N}$ of the plane according to R_H and R_V (and not just the $2^n \times 2^n$ square, for $n = |\mathcal{T}|$, as in the NExpTime case). Equivalently, it suffices to enforce a structure over unary relational names Tiled_t for $t \in \mathcal{T}$, and binary relational names Hor and Ver, such that

1. any element has outgoing Hor and Ver edges such that traversing a Hor edge and then a Ver edge leads to the same element as traversing first a Ver edge and then a Hor edge;
2. relational names Hor and Ver are functional;
3. each element is assigned with at least one relational name Tiled_t; and
4. the assignment of Tiled_t is unique and agrees with R_H and R_V.

Here, a relational name P is *functional* in an instance $\mathcal{D}_0$ if for no element a there are distinct a_1 and a_2 with $\mathsf{P}(a, a_1)$ and $\mathsf{P}(a, a_2)$ in $\mathcal{D}_0$.

Let $(\mathcal{T}, R_H, R_V)$ be a tiling instance. We show how to construct GAV mappings $\mathcal{M}$ with source and global arities bounded by 2, and a Boolean UCQ p over the source schema such that there is a (possibly infinite) source instance $\mathcal{D}$ with $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ if and only if it is possible to tile an infinite plane with the tiling instance (note that here we abuse notation by extending the range of the second and the third arguments of Comply). Intuitively, the only possibility for any such source instance will be to "represent" the tiling of the plane.

We construct mappings $\mathcal{M}$ and query p in several steps, proving in parallel the properties of the corresponding components. Query p will in fact be a conjunction of Boolean CQs and UCQs, and its transformation to a single UCQ is standard. To avoid multiple indexes, we will use the same variables in the conjuncts of p. Since we also omit existential quantifiers as usual, the reader should keep in mind that the variables are local for the conjuncts.

We first show how to enforce the existence of a "square of successors" for each element. Let the first block $\mathcal{M}_1$ of $\mathcal{M}$ be

$$\begin{aligned} \mathsf{Hor}(x,y) &\rightarrow \mathsf{Hor}'(x,y), \\ \mathsf{Ver}(x,y) &\rightarrow \mathsf{Ver}'(x,y), \\ \mathsf{ConfCh}(x) \wedge \mathsf{Hor}(x,y) &\rightarrow \mathsf{ConfCh}^*(), \end{aligned}$$

where Hor and Ver are source binary relational names responsible for horizontal and vertical successors, respectively, Hor' and Ver' are their copies in the global schema, while ConfCh and ConfCh^* are special "challenge" relational names. Let the first conjunct of p be defined as follows (recall that all the variables are implicitly existentially quantified):

$$p_1 = \mathsf{ConfCh}(x) \wedge \mathsf{Hor}(x,y) \wedge \mathsf{Ver}(x,z) \wedge \mathsf{Hor}(z,u) \wedge \mathsf{Ver}(y,u) \wedge \mathsf{Hor}(y,v) \wedge \mathsf{Hor}(u,w).$$

Claim 31. *Let $\mathcal{D}$ be a possibly infinite source instance such that $\mathcal{D} \models p_1$. Then,* $\textsc{Comply}(\emptyset, \mathcal{M}_1, \mathcal{D}, p_1) = \texttt{false}$ *if and only if for each fact $\mathsf{Hor}(a_x, a)$ in $\mathcal{D}$ there are also facts*

$$\mathsf{Hor}(a_x, a_y), \mathsf{Ver}(a_x, a_z), \mathsf{Hor}(a_z, a_u), \mathsf{Ver}(a_y, a_u), \mathsf{Hor}(a_y, a_v), \mathsf{Hor}(a_u, a_w) \tag{12}$$

in $\mathcal{D}$ for some constants a_y, a_z, a_u, a_v and a_w.

Proof. We start with the forward direction. Since $\mathcal{D} \models p_1$, we have that $\mathcal{M}_1 \cup \mathcal{D} \models \mathsf{ConfCh}^*()$. On the one hand, all source instances $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ differ from $\mathcal{D}$ only in the precise element a_x such that $\mathsf{Hor}(a_x, a)$ and $\mathsf{ConfCh}(a_x)$ (and, possibly, some irrelevant atoms). On the other, since $\textsc{Comply}(\emptyset, \mathcal{M}_1, \mathcal{D}, p_1) = \texttt{false}$, for all indistinguishable $\mathcal{D}'$ it holds that $\mathcal{D}' \models p_1$. Therefore, each element a_x with $\mathsf{Hor}(a_x, a)$ in $\mathcal{D}'$ for some a must also have atoms (12) in $\mathcal{D}'$ as required.

We now show the backward direction. Again, since $\mathcal{D} \models p_1$, $\mathcal{M}_1 \cup \mathcal{D} \models \mathsf{ConfCh}^*()$. Thus, each indistinguishable $\mathcal{D}'$ has atoms $\mathsf{Hor}(a_x, a)$ and $\mathsf{ConfCh}(a_x)$ for some a_x. Since it also contains atoms (12) for a_x, we have that $\mathcal{D}' \models p_1$, as required. □

Next, we enforce functionality of Hor and Ver, which together with the previous property guarantees that they form a grid-like structure. Let the second block $\mathcal{M}_2$ of mappings in $\mathcal{M}$ be

$$\begin{aligned} \mathsf{Hor}(x,y_1) \wedge \mathsf{Hor}(x,y_2) \wedge \mathsf{HorCh}_1(y_1) \wedge \mathsf{HorCh}_2(y_2) &\rightarrow \mathsf{H}^*(), \\ \mathsf{Ver}(x,y_1) \wedge \mathsf{Ver}(x,y_2) \wedge \mathsf{VerCh}_1(y_1) \wedge \mathsf{VerCh}_2(y_2) &\rightarrow \mathsf{V}^*(), \end{aligned}$$

where HorCh_i, VerCh_i, H^* and V^* are again special "challenge" relational names. Let the second conjunct of p be defined as follows:

$$p_2 = \mathsf{Hor}(x_1,y_1) \wedge \mathsf{HorCh}_1(y_1) \wedge \mathsf{HorCh}_2(y_1) \wedge \mathsf{Ver}(x_2,y_2) \wedge \mathsf{VerCh}_1(y_2) \wedge \mathsf{VerCh}_2(y_2).$$

The intuition is that the "challenge" relational names represent a test of a pair of a_x, a_{y_1} and a_x, a_{y_2} that are both in the Hor relation or both in the Ver relation. By setting the "challenge" relational names to only these particular a_{y_1} and a_{y_2} we get an indistinguishable instance, and non-compliance would imply that this instance must satisfy the query, which would imply that $a_{y_1} = a_{y_2}$.

Claim 32. *Let $\mathcal{D}$ be a possibly infinite source instance such that $\mathcal{D} \models p_1 \wedge p_2$. Then,* $\textsc{Comply}(\emptyset, \mathcal{M}_1 \cup \mathcal{M}_2, \mathcal{D}, p_1 \wedge p_2) = \texttt{false}$ *if and only if*

- *the condition in Claim 31 holds; and*
- Hor *and* Ver *are functional in $\mathcal{D}$.*

Proof. Suppose $\textsc{Comply}(\emptyset, \mathcal{M}_1 \cup \mathcal{M}_2, \mathcal{D}, p_1 \wedge p_2) = \texttt{false}$ and one of the conditions above fails. By the previous claim we see that it cannot be the first item. If Hor is not functional then for some a_x and distinct a_{y_1}, a_{y_2} we have that $\mathcal{D}$ contains $\mathsf{Hor}(a_x, a_{y_1})$ and $\mathsf{Hor}(a_x, a_{y_2})$. We create $\mathcal{D}'$ by letting HorCh_1 hold only of a_{y_1} and HorCh_2 hold only of a_{y_2}. Thus $\mathcal{D}'$ does not satisfy p_2 (and will not satisfy p_1 as before). The second item implies that $\mathcal{D}$ will have $\mathsf{H}^*()$ in its virtual instance, and the first mapping will imply that $\mathcal{D}'$ will have $\mathsf{H}^*()$ as well. Using this, we can see that $\mathcal{D}'$ and $\mathcal{D}$ are indistinguishable, contradicting the hypothesis that $\textsc{Comply}(\emptyset, \mathcal{M}_1 \cup \mathcal{M}_2, \mathcal{D}, p_1 \wedge p_2) = \texttt{false}$. The case of Ver is argued similarly.

Assume now that the conditions above hold, but $\textsc{Comply}(\emptyset, \mathcal{M}_1 \cup \mathcal{M}_2, \mathcal{D}, p_1 \wedge p_2) = \texttt{true}$ with a witness $\mathcal{D}'$ indistinguishable from $\mathcal{D}$. Indistinguishability of $\mathcal{D}'$ and $\mathcal{D}$ coupled with the fact that $\mathcal{D} \models p_1 \wedge p_2$ imply that $\mathcal{D}'$ contains constants a_x, a_{y_1}, and a_{y_2} with $\mathsf{Hor}(a_x, a_{y_1})$, $\mathsf{Hor}(a_x, a_{y_2})$, $\mathsf{HorCh}_1(a_{y_1})$, and $\mathsf{HorCh}_2(a_{y_2})$ as well as constants b_x, b_{y_1}, and b_{y_2} with $\mathsf{Ver}(b_x, b_{y_1})$, $\mathsf{Ver}(b_x, b_{y_2})$, $\mathsf{VerCh}_1(b_{y_1})$, and $\mathsf{VerCh}_2(b_{y_2})$. Since $\mathcal{D}'$ cannot satisfy p_2 we have either $a_{y_1} \neq a_{y_2}$ or $b_{y_1} \neq b_{y_2}$; either of these contradicts the second item of the claim. □

The immediate consequence is that a non-compliant possibly infinite $\mathcal{D}$ contains a positive quadrant of a plane—that is, there exists a homomorphism from the infinite grid on Hor and Ver with a start point to $\mathcal{D}$ (note that the nodes of the grid do not need to be unique—that is, $\mathcal{D}$ is not necessarily infinite).

The next step is to ensure that each grid node is assigned with a tile type. Let the third block $\mathcal{M}_3$ of $\mathcal{M}$ consist of the mappings

$$\begin{array}{rcll} \mathsf{Tiled}_t(x) & \to & \mathsf{Tiled}'_t(x), & \text{for all } t \in \mathcal{T}, \\ \mathsf{TileCh}(x) \wedge \mathsf{Hor}(x,y) & \to & \mathsf{TileCh}^*(\,), & \end{array}$$

for unary tiling relational names Tiled_t in the source schema, their copies Tiled'_t in the global schema, and "challenge" relational names TileCh and TileCh^*. Let the third conjunct of p be defined as follows:

$$p_3 = \mathsf{TileCh}(x) \wedge \mathsf{Hor}(x,y) \ \wedge \left(\bigvee_{t \in \mathcal{T}} \mathsf{Tiled}_t(x) \right).$$

Intuitively, TileCh is a relational name "challenging" that a particular node has some tile type.

Claim 33. *Let $\mathcal{D}$ be a possibly infinite source instance such that $\mathcal{D} \models p_1 \wedge p_2 \wedge p_3$. Then,* $\textsc{Comply}(\emptyset, \mathcal{M}_1 \cup \mathcal{M}_2 \cup \mathcal{M}_3, \mathcal{D}, p_1 \wedge p_2 \wedge p_3) = \texttt{false}$ *if and only if*

- *the conditions in Claim 32 hold (including the one from Claim 31); and*
- *for every a_x such that $\mathsf{Hor}(a_x, a_y)$ is in $\mathcal{D}$ for some a_y there is $t \in \mathcal{T}$ with $\mathsf{Tiled}_t(a_x)$ in $\mathcal{D}$.*

Proof. Assume $\textsc{Comply}(\emptyset, \mathcal{M}_1 \cup \mathcal{M}_2 \cup \mathcal{M}_3, \mathcal{D}, p_1 \wedge p_2 \wedge p_3)$ is $\texttt{false}$. The first item holds as before. To see the second item, given $\mathsf{Hor}(a_x, a_y) \in \mathcal{D}$ we modify $\mathcal{D}$ to $\mathcal{D}'$ by letting TileCh hold only on a_x. Then $\mathcal{D}'$ is indistinguishable from $\mathcal{D}$, and hence satisfies p_3 by assumption on $\mathcal{D}$, which is only possible if $\mathsf{Tiled}_t(a_x)$ holds in $\mathcal{D}$.

Assume now that $\mathcal{D}$ satisfies the properties in the claim and consider any $\mathcal{D}'$ indistinguishable from $\mathcal{D}$. The fact that $\mathcal{D} \models p_1 \wedge p_2 \wedge p_3$ implies that $\mathsf{TileCh}^*(\,)$ is exported from $\mathcal{D}$, and hence must be exported from $\mathcal{D}'$ as well. Thus there are a_x and a_y in $\mathcal{D}'$ satisfying $\mathsf{TileCh}(a_x) \wedge \mathsf{Hor}(a_x, a_y)$. Further since in the other mappings Hor is exported, we must have $\mathsf{Hor}(a_x, a_y)$ holding in $\mathcal{D}$ as well. By the second item $\mathsf{Tiled}_t(a_x)$ holds in $\mathcal{D}$ for some $t \in \mathcal{T}$. Since other mappings export Tiled_t, we know that $\mathsf{Tiled}_t(a_x)$ holds in $\mathcal{D}'$, which guarantees that p_3 holds in $\mathcal{D}'$ as required. □

The final step is to guarantee that no constant is assigned with two different tile types and the assignment is compatibility-preserving. Let the forth block $\mathcal{M}_4$ of $\mathcal{M}$ be

$$\begin{array}{rcll} \mathsf{OverlapCh}(\,) & \to & \mathsf{OverlapCh}^*(\,), & \\ \mathsf{Tiled}_{t_1}(x) \wedge \mathsf{Tiled}_{t_2}(x) & \to & \mathsf{OverlapCh}^*(\,), & \text{for all } t_1, t_2 \in \mathcal{T} \text{ such that } t_1 \neq t_2, \\ \mathsf{Tiled}_{t_1}(x) \wedge \mathsf{Hor}(x,y) \wedge \mathsf{Tiled}_{t_2}(y) & \to & \mathsf{OverlapCh}^*(\,), & \text{for all } t_1, t_2 \in \mathcal{T} \text{ such that } (t_1, t_2) \notin R_H, \\ \mathsf{Tiled}_{t_1}(x) \wedge \mathsf{Ver}(x,y) \wedge \mathsf{Tiled}_{t_2}(y) & \to & \mathsf{OverlapCh}^*(\,), & \text{for all } t_1, t_2 \in \mathcal{T} \text{ such that } (t_1, t_2) \notin R_V, \end{array}$$

for "challenge" relational names $\mathsf{OverlapCh}$ and $\mathsf{OverlapCh}^*$, and let the fourth conjunct of p be defined as

$$p_4 = \mathsf{OverlapCh}(\,).$$

Finally, let $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2 \cup \mathcal{M}_3 \cup \mathcal{M}_4$ and $p = p_1 \wedge p_2 \wedge p_3 \wedge p_4$.

Claim 34. *Let $\mathcal{D}$ be a possibly infinite source instance such that $\mathcal{D} \models p$. Then,* $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ *if and only if*

- *the conditions in Claim 33 hold; and*
- *the assignment of tile types in $\mathcal{D}$ by relational names Tiled_t is unique and agrees with compatibility relations R_H and R_V.*

Proof. Assume that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ and one of the conditions in the claim fails. By Claims 31–33, it must be the second one. Then, there is a constant assigned multiple tile types in $\mathcal{D}$ or an edge (horizontal or vertical) with an incompatible assignment. Since $\mathcal{D} \models p$, it contains $\mathsf{OverlapCh}(\,)$. Create $\mathcal{D}'$ from $\mathcal{D}$ by removing $\mathsf{OverlapCh}(\,)$. The mappings imply that $\mathsf{OverlapCh}^*(\,)$ holds in the virtual image of $\mathcal{D}$, and from this we have that $\mathcal{D}'$ is indistinguishable from $\mathcal{D}$. This contradicts $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$.

In the other direction, suppose that the conditions hold, and, for the sake of contradiction, that $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{true}$ with a witnessing source instance $\mathcal{D}'$. The fact $\mathsf{OverlapCh}(\,)$ must not be in $\mathcal{D}'$, but $\mathcal{D} \models p$ implies that $\mathsf{OverlapCh}^*(\,)$ is in the virtual image. Thus in $\mathcal{D}'$ one of the other mappings leading to $\mathsf{OverlapCh}^*(\,)$ must trigger, implying a contradiction. □

We conclude that there exists a possibly infinite source instance $\mathcal{D}$ with $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p) = \texttt{false}$ if and only if it is possible to tile an infinite plane with the tiling instance—that is, data-independent compliance is undecidable for UCQs as policies over possibly infinite instances.

Next, we show how to modify this construction to make p be a Boolean CQ. Note that disjunction is used in p only in the third conjunct p_3 to guarantee that each node is assigned a tile type, so we only need to explain how to eliminate this disjunction.

To this end, we first extend $\mathcal{M}_1$ and p_1 to guarantee that, for each fact $\mathsf{Hor}(a_x, a)$ in a non-compliant $\mathcal{D}$ constant a_x has an attached "path gadget": for each tile type t, a_x has a connection to a constant e_t that in turn connects back to constant with all types besides t. Thus if a_x is itself tiled, then it has a connection to an element e that connects back to elements with all tile types. By arranging this we convert the disjunctive property that a_x has some tile type to the conjunctive property that a_x connects up and back to elements of all tile types. The details of this are subtle since we must ensure that there are no "spurious occurrences" of the desired conjunctive property.

For the first step, let $\mathcal{M}'_1$ extend $\mathcal{M}_1$ with the mappings

$$\begin{aligned}
\mathsf{GadgetConnect}_t(x, y) &\rightarrow \mathsf{GadgetConnect}'_t(x, y), && \text{for all } t \in \mathcal{T},\\
\mathsf{Tiled}_t(x) &\rightarrow \mathsf{Tiled}'_t(x), && \text{for all } t \in \mathcal{T},\\
\mathsf{Hor}(x, y) \wedge \mathsf{GadgetConnect}_t(x, z) \wedge \mathsf{GadgetConnect}_s(u, z) \wedge \mathsf{STileCh}(u, z) &\rightarrow \mathsf{STileCh}'(u, z), && \text{for all } t, s \in \mathcal{T}.
\end{aligned}$$

Here $\mathsf{GadgetConnect}_t$ are the connection relational names in the source schema, $\mathsf{GadgetConnect}'_t$ are their copies in the global schema, while $\mathsf{STileCh}(u, z)$ and $\mathsf{STileCh}'(u, z)$ are "challenge" relational names. Note that the second set of mappings in $\mathcal{M}'_1$ are a part of $\mathcal{M}_3$; we will see later that the modification $\mathcal{M}'_3$ of $\mathcal{M}_3$ does not include this set.

Let p'_1 be the Boolean CQ formed by conjoining all the atoms of p_1 and

$$\bigwedge_{t \in \mathcal{T}} \left(\mathsf{GadgetConnect}_t(x, x_t) \wedge \bigwedge_{s \in \mathcal{T} \setminus \{t\}} \left(\mathsf{GadgetConnect}_s(x_t^s, x_t) \wedge \mathsf{STileCh}(x_t^s, x_t) \wedge \mathsf{Tiled}_s(x_t^s) \right) \right).$$

The following claim establishes that in non-compliant instances each element has the "path gadget" attached to it.

Claim 35. *Let $\mathcal{D}$ be a possibly infinite source instance such that $\mathcal{D} \models p'_1$. Then, $\textsc{Comply}(\emptyset, \mathcal{M}'_1, \mathcal{D}, p'_1) = \texttt{false}$ if and only if for each fact $\mathsf{Hor}(a_x, a)$ in $\mathcal{D}$ there are also facts*

$$\begin{aligned}
&\mathsf{Hor}(a_x, a_y), \mathsf{Ver}(a_x, a_z), \mathsf{Hor}(a_z, a_u), \mathsf{Ver}(a_y, a_u), \mathsf{Hor}(a_y, a_v), \mathsf{Hor}(a_u, a_w),\\
&\mathsf{GadgetConnect}_t(a_x, a_{x_t}), \mathsf{GadgetConnect}_s(a_{x_t^s}, a_{x_t}), \mathsf{STileCh}(a_{x_t^s}, a_{x_t}), \mathsf{Tiled}_s(a_{x_t^s}), \qquad \text{for all } t \in \mathcal{T} \text{ and } s \in \mathcal{T} \setminus \{t\},
\end{aligned}$$

in $\mathcal{D}$ for some constants a_y, a_z, a_u, a_v, a_w, as well as a_{x_t} and $a_{x_t^s}$ for all $t, s \in \mathcal{T}$ such that $s \neq t$.

Note that the first set of facts is as in Claim 31.

Proof. We start with the forward direction. Since $\mathcal{D} \models p'_1$, and $\mathcal{M}'_1$ includes $\mathcal{M}_1$, the virtual image of $\mathcal{D}$ satisfies $\mathsf{ConfCh}^*()$. All source instances $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ differ from $\mathcal{D}$ only in the precise element a_x such that $\mathsf{Hor}(a_x, a)$ and $\mathsf{ConfCh}(a_x)$ (and, possibly, some atoms that will be irrelevant for the remainder of the argument). Since $\textsc{Comply}(\emptyset, \mathcal{M}'_1, \mathcal{D}, p'_1) = \texttt{false}$, for all indistinguishable $\mathcal{D}'$, we know that $\mathcal{D}' \models p'_1$. Therefore, keeping in mind that p'_1 contains all atoms of p_1, each element a_x with $\mathsf{Hor}(a_x, a)$ in $\mathcal{D}'$ for some a must also have the required atoms in $\mathcal{D}'$.

Next we show the backward direction. Again, since $\mathcal{D} \models p'_1$, $\mathcal{M}'_1 \cup \mathcal{D} \models \mathsf{ConfCh}^*()$. Therefore, each indistinguishable $\mathcal{D}'$ contains atoms $\mathsf{Hor}(a_x, a)$ and $\mathsf{ConfCh}(a_x)$ for some a_x and a. Since it also contains the required atoms ($\mathsf{GadgetConnect}_s$ atoms and the confluence-related atoms) for this a_x, we have that $\mathcal{D}' \models p'_1$, as required. □

Next, we use these additions to guarantee that every a_x with $\mathsf{Hor}(a_x, a)$ is indeed assigned with a type. Let $\mathcal{M}'_3$ be

$$\begin{aligned}
\mathsf{Hor}(x, y) \wedge \mathsf{GadgetConnect}_t(x, z) \wedge \mathsf{GadgetConnect}_t(x', z) &\rightarrow \mathsf{STileCh}'(x', z), \text{ for all } t \in \mathcal{T},\\
\mathsf{Hor}(x, y) \wedge \mathsf{STileCh}(x, z) &\rightarrow \mathsf{STileCh}^*(),
\end{aligned}$$

and let p'_3 be the Boolean CQ

$$p'_3 = \bigwedge_{s \in \mathcal{T}} \left(\mathsf{GadgetConnect}_s(x_s, x) \wedge \mathsf{STileCh}(x_s, x) \wedge \mathsf{Tiled}_s(x_s) \right).$$

Claim 36. *Let $\mathcal{D}$ be a possibly infinite source instance such that $\mathcal{D} \models p'_1 \wedge p_2 \wedge p'_3$. Then, $\textsc{Comply}(\emptyset, \mathcal{M}'_1 \cup \mathcal{M}_2 \cup \mathcal{M}'_3, \mathcal{D}, p'_1 \wedge p_2 \wedge p'_3) = \texttt{false}$ if and only if*

– *the condition in Claim 35 and the second condition in Claim 32 hold; and*

– *for every a_x such that* $\mathsf{Hor}(a_x, a) \in \mathcal{D}$ *for some a there is $t \in \mathcal{T}$ with* $\mathsf{Tiled}_t(a_x) \in \mathcal{D}$.

Proof. Assume that $\textsc{Comply}(\emptyset, \mathcal{M}'_1 \cup \mathcal{M}_2 \cup \mathcal{M}'_3, \mathcal{D}, p'_1 \wedge p_2 \wedge p'_3)$ is `false`. The first item in the claim holds as before, so we focus on the second item. Given $\mathsf{Hor}(a_x, a) \in \mathcal{D}$ we modify $\mathcal{D}$ to $\mathcal{D}'$ as follows:

1. we remove every fact of the form $\mathsf{STileCh}(a_{x'}, a_z)$ such that

$$\bigvee_{t \in \mathcal{T}} \exists x.\, \exists y.\, \mathsf{Hor}(x, y) \wedge \mathsf{GadgetConnect}_t(x, a_z) \wedge \mathsf{GadgetConnect}_t(a_{x'}, a_z)$$

   holds in $\mathcal{D}$;

2. we add every fact $\mathsf{STileCh}(a_x, a_z)$, for any a_z, such that $\mathsf{GadgetConnect}_s(a_x, a_z)$ is in $\mathcal{D}$ for $s \in \mathcal{T}$; and

3. we remove every fact $\mathsf{STileCh}(a_{x'}, a_z)$ not added in the previous step, such that

$$\bigwedge_{t,s \in \mathcal{T}, t \neq s} \left(\neg \exists x.\, \exists y.\, \mathsf{Hor}(x, y) \wedge \mathsf{GadgetConnect}_t(x, a_z) \wedge \mathsf{GadgetConnect}_s(a_{x'}, a_z)\right)$$

   holds in $\mathcal{D}$.

We first claim that $\mathcal{D}'$ is indistinguishable from $\mathcal{D}$. The mapping $\mathsf{Hor}(x, y) \wedge \mathsf{STileCh}(x, z) \rightarrow \mathsf{STileCh}^*()$ is clearly not impacted by any of the changes above, since the second item ensures that it is still applicable. We may worry about the impact of all the changes above on $\mathcal{M}'_1$—that is, in particular, whether the effect of the following mapping is not changed:

$$\mathsf{Hor}(x, y) \wedge \mathsf{GadgetConnect}_t(x, z) \wedge \mathsf{GadgetConnect}_s(u, z) \wedge \mathsf{STileCh}(u, z) \rightarrow \mathsf{STileCh}'(u, z).$$

However, the removal in the first step does not have any impact, since although we are removing elements where this mapping is applicable, all these elements also trigger one of the following mappings in the first set of $\mathcal{M}'_3$:

$$\mathsf{Hor}(x, y) \wedge \mathsf{GadgetConnect}_t(x, z) \wedge \mathsf{GadgetConnect}_t(x', z) \rightarrow \mathsf{STileCh}'(x', z),$$

so the exported information is the same. Similarly, the addition in the second step does not have any impact, since $\mathsf{STileCh}'(a_x, a_z)$ was already exported due to the same mappings in $\mathcal{M}'_3$. Finally, the removal in the third step does not have any impact, since by definition we are removing elements where the mapping did not fire.

Hence $\mathcal{D}'$ satisfies p'_3 by the assumption on $\mathcal{D}$. Thus there are (not necessary different) a_c and a_s, for $s \in \mathcal{T}$, such that

$$\bigwedge_{s \in \mathcal{T}} (\mathsf{GadgetConnect}_s(a_s, a_c) \wedge \mathsf{STileCh}(a_s, a_c) \wedge \mathsf{Tiled}_s(a_s))$$

holds in $\mathcal{D}'$. In particular all $\mathsf{Tiled}_s(a_s)$ hold in $\mathcal{D}$ as well, since $\mathcal{D}'$ agrees with $\mathcal{D}$ on the relations Tiled_s. We claim that one of these a_s must be a_x, which would suffice to prove the forward direction of the claim. We first note that no a_s other than a_x can be in the first position of Hor. This is because any other element in that position would have been removed from $\mathsf{STileCh}$ in the first removal step above, and would not have been replaced. Thus it suffices to argue that there is an a_s that is in the first position of Hor. Now for each a_s other than a_x there are $a_{x'}$ and $a_{y'}$ such that $\mathsf{Hor}(a_{x'}, a_{y'}) \wedge \mathsf{GadgetConnect}_{t'}(a_{x'}, a_c) \wedge \mathsf{GadgetConnect}_{s'}(a_s, a_c)$ holds in $\mathcal{D}'$ for some $s', t' \in \mathcal{T}$, since otherwise the pair (a_s, a_c) would have been removed at the third step. Take one of these elements a_s different from a_x (if all of them equal a_x, then the claim is automatic), and consider the corresponding t' and $a_{t'}$. If $a_{t'}$ is not a_x, then we would have removed $\mathsf{STileCh}(a_{t'}, a_c)$ in $\mathcal{D}'$ in the first removal step above, which is a contradiction. Hence we have $a_{t'}$ is a_x, and we are done.

In the other direction, suppose $\mathcal{D}$ satisfies the properties above and consider any $\mathcal{D}'$ indistinguishable from $\mathcal{D}$. The fact that $\mathcal{D} \models p'_1 \wedge p_2 \wedge p'_3$ implies that $\mathsf{STileCh}^*()$ is exported from $\mathcal{D}$, and hence must be exported from $\mathcal{D}'$. Thus there are a_x, a_y, and a_z in $\mathcal{D}'$ satisfying $\mathsf{Hor}(a_x, a_y) \wedge \mathsf{STileCh}(a_x, a_z)$. Further since in the other mappings Hor is exported, we must have $\mathsf{Hor}(a_x, a_y)$ holding in $\mathcal{D}$ as well. By the second item $\mathsf{Tiled}_t(a_x)$ holds in $\mathcal{D}$ for some $t \in \mathcal{T}$. Since other mappings export Tiled_t, we know that $\mathsf{Tiled}_t(a_x)$ holds in $\mathcal{D}'$, which guarantees that p'_3 is in $\mathcal{D}'$ as required. □

Let $\mathcal{M}' = \mathcal{M}'_1 \cup \mathcal{M}_2 \cup \mathcal{M}'_3 \cup \mathcal{M}_4$ and $p' = p'_1 \wedge p_2 \wedge p'_3 \wedge p_4$. We conclude that there exists a possibly infinite $\mathcal{D}$ with $\textsc{Comply}(\emptyset, \mathcal{M}', \mathcal{D}, p') =$ `false` if and only if the tiling instance has a solution; thus, the problem is undecidable over possibly infinite instances.

Recall that infinite source instances are not allowed in the general settings of this paper. Next we explain how to make the proof work for the case when only finite source instances are considered. We start with a definition and a known fact.

A solution of a tiling instance $(\mathcal{T}, R_H, R_V)$ is *periodic* if there exist positive numbers n and m such that the tile types assigned to (i, j), $(n + i, j)$ and $(i, m + j)$ are the same for all $i, j \in \mathbb{N}$. A periodic tiling can be seen as a tiling of a torus, since column $n + 1$ and row $m + 1$ can be "glued" with the left-most column and bottom row, respectively.

Let S_{tiling} denote the set of all tiling instances that allow for solutions, and S_{period} denote the set of all tiling instances that allow for periodic solutions. Our proof is based on the following fact.

Fact 37 ([17, 42])**.** *Sets* S_{tiling} *and* S_{period} *are* recursively inseparable*—that is, there is no recursive set* S *of tiling instances such that* $S_{\text{period}} \subseteq S \subseteq S_{\text{tiling}}$.

To use this fact for undecidability of ComplyAll over finite source instances, it is enough to show that

1. if a tiling instance $(\mathcal{T}, R_H, R_V)$ is in S_{period} then $\text{ComplyAll}(\emptyset, \mathcal{M}, p) = \texttt{false}$ for $\mathcal{M}$ and p based on $(\mathcal{T}, R_H, R_V)$ as in the reduction above;
2. if $\text{ComplyAll}(\emptyset, \mathcal{M}, p) = \texttt{false}$ for $\mathcal{M}$ and p based on a tiling instance $(\mathcal{T}, R_H, R_V)$ as in the reduction above then $(\mathcal{T}, R_H, R_V) \in S_{\text{tiling}}$.

In fact, item 2 follows immediately from the correctness of the reduction for possibly infinite source instances. Item 1 is also straightforward: since the tiling is periodic, the infinite source instance witnessing data-independent compliance in the reduction can be easily transformed to a finite torus-like source instance by appropriately identifying constants in the grid. □

The proof of Theorem 30 relies on the fact that different mappings may use the same relational names in the head—that is, they are not CQ views. In the next section we will see that this is critical, and for CQ views the problems are decidable (provided the ontology is empty as before). However, the effect of GAV mappings with the same head relational name can be easily simulated by means of full linear TGDs of the form $\mathsf{R}_1(\mathbf{x}) \rightarrow \mathsf{R}_2(\mathbf{x})$ for any global relational names R_1 and R_2 of the same arity.

Corollary 38. *Problem* $\text{ComplyAll}(\mathcal{O}, \mathcal{M}, p)$ *is undecidable if* $\mathcal{O}$ *ranges over an ontology language containing all sets of full linear TGDs,* $\mathcal{M}$ *ranges over a mapping language containing all sets of mappings that are constant-free, CQ views and of source and global arities bounded by 2, and* p *ranges over a policy language containing all constant-free Boolean CQs.*

In the passing, we conjecture that the undecidability results in Theorem 30 and Corollary 38 extend to the setting of StrComplyAll. However, the current construction is not sufficient, since when no tiling exists, the negation of p may still be revealed. Ideally, one would modify the construction so that the negation of p is never revealed regardless of whether a tiling exists. However, this has proved to be rather challenging technically, and we leave it for future work.

7.2. Decidable Cases

We now complement the picture for data-independent compliance by showing that the problem is decidable when the mappings are restricted to CQ views and there is no ontology.

We exploit the *critical instance* method, which has been used to establish both decidability and undecidability results for a number of reasoning problems [4, 7, 28, 39, 63, 70]. The idea in our setting is to reduce data-independent compliance checking to checking the usual data-dependent notion over a single "canonical" instance. In particular, we show that if there is any non-compliant source instance, then the *critical instance* for the source relation names mentioned in the mappings is also a witness to non-compliance.

Definition 39. *The* critical instance $C_{\mathbf{R}}$ *for a schema (i.e., set of relational names)* $\mathbf{R}$ *is the instance whose domain consists of a single constant* $*$ *and whose facts are* $\mathsf{R}(*, \ldots, *)$ *for all relational names* R *in* $\mathbf{R}$.

Note that every Boolean policy over the relevant relations holds on the critical instance for the source schema, and thus it is intuitively the "hardest" instance to get to comply. The key property of the critical instance is established by the following lemma.

Lemma 40. *Let* $\mathcal{M}$ *be a set of constant-free CQ views with source schema* $\mathbf{S}$ *and* p *be a constant-free Boolean policy. Then,* $\text{ComplyAll}(\emptyset, \mathcal{M}, p) = \texttt{true}$ *if and only if* $\text{Comply}(\emptyset, \mathcal{M}, C_{\mathbf{S}}, p) = \texttt{true}$.

Proof. The proof of the lemma relies on a more general result established in [7], which we formulate next.

Let $\mathbf{G}$ denote the global schema (without loss of generality, we assume that for each relational name in $\mathbf{G}$ there exists a mapping in $\mathcal{M}$ with this relational name in the head). Let $\mathcal{T}$ be a set of TGDs over both $\mathbf{S}$ and $\mathbf{G}$; in particular, $\mathcal{T}$ might contain TGDs relating source and global relations, as well as TGDs over source or over global schema only. Given an instance $\mathcal{G}$ over $\mathbf{G}$, an instance $\mathcal{F}$ over both $\mathbf{S}$ and $\mathbf{G}$ is a *realisation* of $\mathcal{G}$ if $\mathcal{F}$ satisfies $\mathcal{T}$ and the restriction of $\mathcal{F}$ to $\mathbf{G}$ is $\mathcal{G}$. Instance $\mathcal{G}$ is *realisable* for $\mathcal{T}$ if it admits a realisation. For a Boolean CQ p over both schemata, $\text{Comply}^G(\mathcal{T}, \mathcal{G}, p)$ is `true` if and only if there is a realisation $\mathcal{F}$ satisfying $\mathcal{T}$ such that p does not hold in $\mathcal{F}$. Note that $\text{Comply}^G(\mathcal{T}, \mathcal{G}, p)$ is vacuously `false` if $\mathcal{G}$ is not realisable. We use Comply^G here to stress that input instances are over the global schema, while the compliance problems we have studied so far admit source instances as input. Let now $\text{ComplyAll}^G(\mathcal{T}, p)$ be `true` if and only if $\text{Comply}^G(\mathcal{T}, \mathcal{G}, p)$ is `true` for every realisable instance $\mathcal{G}$ over $\mathbf{G}$.

We are now ready to state the result from [7], which establishes that ComplyAll^G can always be reduced to Comply^G over the critical instance for the global schema.

Fact 41 ([7]). *For a set $\mathcal{T}$ of constant-free TGDs and a constant-free Boolean CQ p over both* $\mathbf{S}$ *and* $\mathbf{G}$, $\textsc{ComplyAll}^G(\mathcal{T}, p)$ *is* `true` *if and only if* $\textsc{Comply}^G(\mathcal{T}, C_{\mathbf{G}}, p)$ *is* `true`.

The full proof of this fact can be found in [6, Theorem 10] (note that the result is stated there in terms of the complement of $\textsc{ComplyAll}^G$).

We now discuss how this result can be exploited to show the lemma. Let $\mathcal{T}_{\mathcal{M}}$ extend $\mathcal{M}$ with the *inverse mapping* $\mathsf{R}(\mathbf{x}) \rightarrow \exists \mathbf{z}.\varphi(\mathbf{x}, \mathbf{z})$ for each mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \mathsf{R}(\mathbf{x})$ in $\mathcal{M}$. The following claim relates the compliance problems studied in [7] to those in our paper for the particular case where the input mappings are CQ views.

Claim 42. *For a set of constant-free CQ views $\mathcal{M}$, source instance $\mathcal{D}$, and constant-free Boolean policy p,* $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p)$ *is* `true` *if and only if* $\textsc{Comply}^G(\mathcal{T}_{\mathcal{M}}, \mathcal{V}_{\mathcal{M},\mathcal{D}}, p)$ *is* `true`.

Proof. For the forward direction, suppose $\mathcal{D}'$ witnesses $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p)$—that is, $\mathcal{D}'$ is indistinguishible from $\mathcal{D}$ for $\mathcal{M}$ and satisfies $\neg p$. Let $\mathcal{F}$ be the union of $\mathcal{D}$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}}$. Then, $\mathcal{F}$ witnesses $\textsc{Comply}^G(\mathcal{T}_{\mathcal{M}}, \mathcal{V}_{\mathcal{M},\mathcal{D}}, p)$, as required.

For the backward direction, assume that an instance $\mathcal{F}$ over both $\mathbf{S}$ and $\mathbf{G}$ witnesses $\textsc{Comply}^G(\mathcal{T}_{\mathcal{M}}, \mathcal{V}_{\mathcal{M},\mathcal{D}}, p)$—that is, $\mathcal{F}$ is a realisation of $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ that satisfies $\neg p$. Let $\mathcal{D}'$ be the restriction of $\mathcal{F}$ to $\mathbf{S}$. Then, $\mathcal{D}'$ is indistinguishable from $\mathcal{D}$ for $\mathcal{M}$. Since p mentions only source relational names, $\mathcal{D}' \models \neg p$. Thus $\mathcal{D}'$ witnesses $\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p)$. □

We finally argue that Fact 41 and Claim 42 imply the statement in our theorem:

$\textsc{ComplyAll}(\emptyset, \mathcal{M}, p)$	if and only if, by the definition of $\textsc{ComplyAll}$,
$\textsc{Comply}(\emptyset, \mathcal{M}, \mathcal{D}, p)$ for all source instances $\mathcal{D}$	if and only if, by Claim 42,
$\textsc{Comply}^G(\mathcal{T}_{\mathcal{M}}, \mathcal{V}_{\mathcal{M},\mathcal{D}}, p)$ for all source instances $\mathcal{D}$	if and only if, by the definition of a realisable instance,
$\textsc{Comply}^G(\mathcal{T}_{\mathcal{M}}, \mathcal{G}, p)$ for all realisable global instances $\mathcal{G}$	if and only if, by the definition of $\textsc{ComplyAll}^G$,
$\textsc{ComplyAll}^G(\mathcal{T}_{\mathcal{M}}, p)$	if and only if, by Fact 41,
$\textsc{Comply}^G(\mathcal{T}_{\mathcal{M}}, C_{\mathbf{G}}, p)$	if and only if, by Claim 42,
$\textsc{Comply}(\emptyset, \mathcal{M}, C_{\mathbf{S}}, p)$.	

Note that at the last step we use the assumption that all relational names in $\mathbf{G}$ are mentioned in $\mathcal{M}$. □

From Theorem 19 and Lemma 40, we immediately obtain decidability of $\textsc{ComplyAll}(\emptyset, \mathcal{M}, p)$ in Σ_2^p under the assumption that $\mathcal{M}$ is a set of constant-free CQ views and p is a constant-free Boolean CQ. This upper bound is, however, not tight since we can exploit the special structure of the critical instance to obtain more favourable complexity. By Lemma 40, in this case $\textsc{ComplyAll}(\emptyset, \mathcal{M}, p)$ is equivalent to checking $\textsc{Comply}(\emptyset, \mathcal{M}, C_{\mathbf{S}}, p)$, which requires evaluating p over all the source instances indistinguishable from $C_{\mathbf{S}}$ for $\mathcal{M}$. Using again an idea from [7], we argue that it suffices to evaluate the policy over a single "universal" indistinguishable source instance, as in the following algorithm for $\textsc{ComplyAll}(\emptyset, \mathcal{M}, p)$:

1. set source instance $\mathcal{D}_{\text{univ}}$ to $\emptyset$;
2. for each mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \mathsf{R}(\mathbf{x})$ in $\mathcal{M}$
   - add to $\mathcal{D}_{\text{univ}}$ all facts in $\varphi(h(\mathbf{x}), h(\mathbf{z}))$, for a function h sending all variables $\mathbf{x}$ to $*$ and $\mathbf{z}$ to fresh constants;
3. while there is a mapping $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \mathsf{R}(\mathbf{x})$ in $\mathcal{M}$ and a function h such that $\mathcal{D}_{\text{univ}} \models \varphi(h(\mathbf{x}), h(\mathbf{z}))$ and h maps a variable in $\mathbf{x}$ to a constant different from $*$
   - replace each constant $h(*)$ in all facts in $\mathcal{D}_{\text{univ}}$ with $*$;
4. return `true` if and only if $\mathcal{D}_{\text{univ}} \not\models p$.

The following lemma establishes the universality property of $\mathcal{D}_{\text{univ}}$ constructed in the algorithm, justifying its correctness.

Lemma 43. *Let $\mathcal{M}$ be a set of constant-free CQ views $\mathcal{M}$, p be a constant-free Boolean policy, and $\mathcal{D}_{\text{univ}}$ be the source instance constructed by the algorithm. For any instance $\mathcal{D}'$ indistinguishable from the critical source instance $C_{\mathbf{S}}$ for $\mathcal{M}$, there exists a homomorphism from $\mathcal{D}_{\text{univ}}$ to $\mathcal{D}'$.*

Proof. First, note that, by construction, $\mathcal{D}_{\text{univ}}$ is indistinguishable from $C_{\mathbf{S}}$. Consider now an arbitrary $\mathcal{D}'$ indistinguishible from $C_{\mathbf{S}}$. The source instance $\mathcal{D}_{\text{univ}}$ is the fixpoint of a sequence of structures $\mathcal{D}^i_{\text{univ}}$, where $\mathcal{D}^0_{\text{univ}}$ is $\mathcal{D}_{\text{univ}}$ upon completion of the first cycle, and $\mathcal{D}^{i+1}_{\text{univ}}$ is $\mathcal{D}_{\text{univ}}$ on a step of the second cycle, formed from $\mathcal{D}^i_{\text{univ}}$ by identifying constants with $*$. We show that there is a function h that serves as a homomorphism from each $\mathcal{D}^i_{\text{univ}}$ into $\mathcal{D}'$. Note that in $\mathcal{D}^0_{\text{univ}}$ each constant is either $*$ or a fresh constant generated to satisfy an inverse mapping $\mathsf{R}(\mathbf{x}) \rightarrow \exists \mathbf{z}.\, \varphi(\mathbf{x}, \mathbf{z})$, and this mapping should be satisfied by $\mathcal{D}'$ as well. We let h map $*$ to itself and each fresh constant a_z generated to witness each $z \in \mathbf{z}$ with the corresponding constant witnessing z in $\mathcal{D}'$. Function h is a homomorphism from $\mathcal{D}^0_{\text{univ}}$ to $\mathcal{D}_{\text{univ}}$. Next we show, by induction on i, that h is a homomorphism from each $\mathcal{D}^i_{\text{univ}}$

into $\mathcal{D}'$ as well. In the inductive step, suppose we identify a constant a with $*$ in going from $\mathcal{D}^i_{\text{univ}}$ to $\mathcal{D}^{i+1}_{\text{univ}}$. Then we know that $h(a) = *$ in $\mathcal{D}'$. From this and the inductive hypothesis, we can argue that the new facts added in $\mathcal{D}^{i+1}_{\text{univ}}$ are already preserved by h. □

Theorem 44. *Problem* ComplyAll$(\mathcal{O}, \mathcal{M}, p)$ *for empty* $\mathcal{O}$*,* $\mathcal{M}$ *in a mapping language* $\mathbb{M}$*, and* p *in a policy language* $\mathbb{P}$ *is in*

(1) coNP*, if* $\mathbb{M}$ *consists of constant-free CQ views and* $\mathbb{P}$ *consists of constant-free Boolean policies; and*

(2) AC^0*, if the condition of Case (1) holds and* $\mathbb{M}$ *is also LAV.*

Proof. By Lemmas 40 and 43 we conclude that the algorithm correctly decides ComplyAll$(\emptyset, \mathcal{M}, p)$. Note that the construction of $\mathcal{D}_{\text{univ}}$ can be equivalently described as a monotone process, by first forming an initial set $\mathcal{D}^0_{\text{univ}}$ as in Lemma 43, and then adding at each step new equality facts (corresponding to the equalities described above between freshly introduced constants and the special constant $*$) and new facts (formed via congruence). The process must reach a fixpoint in polynomially many steps since only linearly many fresh constants are introduced in $\mathcal{D}^0_{\text{univ}}$ and in each step at least one such constant is identified with $*$; we abuse notation by letting $\mathcal{D}^0_{\text{univ}}, \ldots, \mathcal{D}^n_{\text{univ}}$ denote this monotone sequence. The initial set of facts $\mathcal{D}^0_{\text{univ}}$ can be formed in polynomial time. For $\mathcal{D}^{i+1}_{\text{univ}}$ we can guess the required homomorphisms that support the new equality and verify both that they are homomorphisms and that all new facts in $\mathcal{D}^{i+1}_{\text{univ}}$ are generated from $\mathcal{D}^i_{\text{univ}}$ using the newly-generated equality and the congruence rules for equality. Since we can also guess a homomorphism from p and verify it, we get the desired coNP bound for *Case (1)*.

Finally, for *Case (2)*, note that if the mappings are LAV, then $\mathcal{D}_{\text{univ}}$ can be constructed in AC^0. Furthermore, in evaluating p, we can leverage that any joined variable of p can hold in $\mathcal{D}_{\text{univ}}$ only on $*$, and thus after substituting this we need only check the individual atoms separately. □

Next we give the matching lower bound to the *Case 1* of Theorem 44.

Proposition 45. *Problem* ComplyAll$(\mathcal{O}, \mathcal{M}, p)$ *is* coNP*-hard if* $\mathcal{O}$ *is empty,* $\mathcal{M}$ *ranges over a mapping language containing all sets of constant-free CQ views of source arity bounded by 2 and global arity bounded by 0, and* p *ranges over a policy language containing all constant-free Boolean CQs.*

Proof. We provide a reduction of graph homomorphism, a canonical NP-hard problem, to the complement of ComplyAll. We employ a variation of an argument in [7] to do this. Given directed graphs G_1 and G_2, let q_1 and q_2 be the corresponding Boolean CQs over the binary relational name Edge. Let the source schema consist of this relational name and the global schema consist of a nullary relational name Good. Let also $\mathcal{M}$ simply state $q_1 \rightarrow \text{Good}()$ and the policy p be q_2. We claim that G_2 has a homomorphism to G_1 exactly when ComplyAll$(\emptyset, \mathcal{M}, p)$ is `false`.

First, suppose there is a non-compliant source instance $\mathcal{D}$. We claim that q_1 must hold on $\mathcal{D}$. Indeed, otherwise the empty instance would be indistinguishable from $\mathcal{D}$, and, since the empty instance clearly satisfies $\neg p$, we obtain a contradiction to the assumption that $\mathcal{D}$ is non-compliant. Thus, non-compliance of $\mathcal{D}$ implies that every $\mathcal{D}'$ having q_1 `true` also has q_2 `true`, which is equivalent to a homomorphism from G_2 to G_1.

Conversely, suppose G_2 has a homomorphism to G_1. Take the source instance representing G_1 as $\mathcal{D}$. Then, Good() will hold on its image, and every $\mathcal{D}'$ indistinguishable from $\mathcal{D}$ must have a homomorphism from q_1 (i.e., G_1), and hence from G_2. □

To conclude this section, we remind that we leave open the questions on decidability and precise complexity of ComplyAll in the settings where the mappings are LAV.

8. Implications of Our Results

In this section we discuss the implications of our work. We first argue that our results correct certain erroneous complexity bounds claimed in [66]. We then show that our results also close an open problem in the area of data pricing in databases [54].

8.1. Privacy in GLAV Information Integration

Nash and Deutsch [66] considered problems similar to ours in the context of data integration via GLAV mappings. In particular, their key privacy guarantee (Guarantee 2 in [66]) coincides, in the case of Boolean policies, with our notion of strong compliance in Section 3.3 with no ontology. In contrast to our work, however, they did not formulate their complexity results in terms of the size of mappings $\mathcal{M}$, source dataset $\mathcal{D}$, and policy p, but rather in terms of the size of $\mathcal{D}$, p, and *rewritings* of the heads of mappings, as defined next.

Definition 46. *Let* $\mathcal{M}$ *be a set of mappings and* q *be a CQ over the global schema. A UCQ* q' *over the source schema is a* rewriting *of* q *over* $\mathcal{M}$ *if, for every source instance* $\mathcal{D}$ *and tuple* $\mathbf{c}$ *of constants,* $\mathcal{D} \models q'(\mathbf{c})$ *if and only if* $\mathbf{c}$ *is a certain answer to* q *over the data integration setting* $(\emptyset, \mathcal{M}, \mathcal{D})$*.*

The decision problem StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m \in \mathcal{M}})$ *is defined exactly as* StrComply$(\emptyset, \mathcal{M}, \mathcal{D}, p)$*, except that there is an additional input that is the rewritings* q_m *over* $\mathcal{M}$ *of the heads of the mappings* m *in* $\mathcal{M}$*.* Data complexity *of problem* StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m \in \mathcal{M}})$ *assumes that* $\mathcal{M}$*,* p*, and* $\{q_m\}_{m \in \mathcal{M}}$ *are fixed, while only* $\mathcal{D}$ *forms the input.*

The result on checking Guarantee 2, established in [66, Corollary 3], can then be reformulated using our notation as follows.

Statement 47 ([66])**.** *Problem* StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m\in\mathcal{M}})$*, for* $\mathcal{M}$ *ranging over all mappings,* p *ranging over Boolean policies, and rewritings* $\{q_m\}_{m\in\mathcal{M}}$ *of the heads of the mappings in* $\mathcal{M}$*, is in* NP.

Note that, if $\mathcal{M}$ consists of CQ views, then the rewriting of the head of any mapping in $\mathcal{M}$ is just its body. Thus, any lower bound for StrComply$(\emptyset, \mathcal{M}, \mathcal{D}, p)$, for $\mathcal{M}$ ranging over sets of CQ views, also applies to StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m\in\mathcal{M}})$. We can hence obtain the following result as a corollary of our Theorems 19 and 24, which contradicts the NP upper bound in Statement 47 and shows that the problem is much harder.

Corollary 48. *Problem* StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m\in\mathcal{M}})$ *for* $\mathcal{M}$ *ranging over sets of CQ views,* p *ranging over Boolean policies, and rewritings* $\{q_m\}_{m\in\mathcal{M}}$ *of the heads of the mappings in* $\mathcal{M}$*, is* NExpTime*-complete.*

Nash and Deutsch also provided a tractable case in data complexity, where they required $\mathcal{M}$ to satisfy the so-called "tagged-union" property (see [66, Theorem 4]). In our notation, their result can be reformulated as follows.

Statement 49 ([66])**.** *Problem* StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m\in\mathcal{M}})$*, for* $\mathcal{M}$ *ranging over sets of mappings having tagged-unions,* p *ranging over Boolean policies, and rewritings* $\{q_m\}_{m\in\mathcal{M}}$ *of the heads of the mappings in* $\mathcal{M}$*, is in* P *in data complexity.*

It is immediate to check that any set of CQ views have tagged-unions. Therefore, Statement 49 is in contradiction with our NP-hardness given in *Case (a)* of Theorem 29, which applies already if $\mathcal{M}$ ranges over sets of mappings that are CQ views and LAV (the matching upper bound follows from *Case (4)* of Theorem 19).

Corollary 50. *Problem* StrComply$^{Rew}(\mathcal{M}, \mathcal{D}, p, \{q_m\}_{m\in\mathcal{M}})$*, for* $\mathcal{M}$ *ranging over sets of mappings having tagged-unions,* p *ranging over Boolean policies, and rewritings* $\{q_m\}_{m\in\mathcal{M}}$ *of the heads of the mappings in* $\mathcal{M}$*, is* NP*-complete in data complexity.*

8.2. Instance-Based Determinacy and Data Pricing

In [54], Koutris et al. study the problem of query-based data pricing, where the goal is to automatically assign a price to a chunk of data (say, in a Web-based marketplace) given the price already assigned to a set of views. The data pricing framework in [54] has been used as a foundation for a growing body of work in the database community [33, 53, 57, 58, 59].

A central element to this framework is the notion of *instance-based determinacy*. Intuitively, a set of CQ views determines a query if we can compute all answers to the query only from the results of given views and without having access to the underpinning data. Using our notation, we can equivalently define instance-based determinacy as follows.

Definition 51. *A set of CQ views* $\mathcal{M}$ determines *a CQ* p *over the source schema given a source instance* $\mathcal{D}$ *if, for each source instance* $\mathcal{D}'$ *such that* $\mathcal{V}_{\mathcal{M},\mathcal{D}'}$ *coincides with* $\mathcal{V}_{\mathcal{M},\mathcal{D}}$*, the answers to* p *over* $\mathcal{D}$ *and* $\mathcal{D}'$ *also coincide.*

Then, Determinacy$(\mathcal{M}, \mathcal{D}, p)$ *is* `true` *if and only if* $\mathcal{M}$ *determines* p *given* $\mathcal{D}$*, while* Determinacy$^{Ext}(\mathcal{M}, \mathcal{D}, p, \mathcal{V}_{\mathcal{M},\mathcal{D}})$ *is the same problem, but with the virtual image* $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ *as an additional parameter.*[2]

Koutris et al. showed in [54] that Determinacy^{Ext} is in Π_2^p, but left the lower bound open. Furthermore, they did not study the (arguably more natural) problem Determinacy.

It is immediate to see that, for the empty ontology and Boolean policies, Determinacy is precisely the complement of StrComply. Thus, our results in Theorems 19, 24, 25 and 26 transfer directly to instance-based determinacy as given next.

Corollary 52. *Problem* Determinacy$(\mathcal{M}, \mathcal{D}, p)$ *is* coNExpTime*-complete when* $\mathcal{M}$ *ranges over sets of CQ views and* p *ranges over Boolean CQs, and the lower bound holds already if* p *ranges over facts. Problem* Determinacy$^{Ext}(\mathcal{M}, \mathcal{D}, p, \mathcal{V}_{\mathcal{M},\mathcal{D}})$ *is* Π_2^p*-hard if either* $\mathcal{M}$ *ranges over a mapping language containing all sets of CQ views of bounded source arity and* p *ranges over a policy language containing all facts, or if* $\mathcal{M}$ *ranges over a mapping language containing all set of mappings that are CQ views, LAV and of bounded source arity, and* p *ranges over a policy language containing all Boolean CQs.*

Our results also significantly refine the coNP lower bounds in data complexity for determinacy given in [54]. In particular, from Theorem 29 we immediately have that Determinacy$(\mathcal{M}, \mathcal{D}, p)$ and Determinacy$^{Ext}(\mathcal{M}, \mathcal{D}, p, \mathcal{V}_{\mathcal{M},\mathcal{D}})$ are both coNP-hard in data complexity if either $\mathcal{M}$ ranges over a mapping language containing all CQ views and p ranges over a policy language containing all facts, or $\mathcal{M}$ ranges over a mapping language containing all sets that are both CQ views and LAV, and p ranges over a policy language containing all Boolean CQs.

[2] The latter is the actual problem considered in [54].

9. Other Related Work

The problem of preventing disclosure of sensitive data in information systems has received significant attention in recent years. The high-level goal in most existing approaches is to release (or provide access to) a dataset to the public in such a way that the data remains maximally accessible to users without compromising the privacy of sensitive information.

Many different models have been proposed in the literature, and they can be roughly divided into the following three kinds.

1. Perturbation models, where (typically numeric) data is modified by the introduction of noise (e.g., in the form of a random variable with suitable properties). In this setting, the goal is for the data to remain useful for statistical analysis, rather than to preserve its integrity. This includes, for instance, work on *differential privacy* [36].

2. Anonymisation and abstraction models, where some data is suppressed (e.g., by replacing constants with generated anonymous identifiers), or generalised (e.g., by replacing a concrete numeric value with a value range) in a way that preserves data integrity. This includes, for instance, work on k-anonymity and related notions in databases [5, 62, 69, 75] as well as work on anonymisation in linked data and ontologies [30, 31].

3. View-based models, where the data made accessible (or inaccessible) to users is declaratively defined by means of views expressed in some logic-based language [7, 24, 26, 64, 78]. In contrast to other approaches, data in view-based models is not modified; instead, access to it is controlled by means of a declaratively specified layer.

We focus the further discussion on view-based models, which are the closest to our work.

Miklau and Suciu in [64] study the problem of whether a set of CQ views $\mathcal{M}$ logically discloses information about a secret CQ q. Although their framework is probabilistic, the key notion of *perfect privacy* has a logical counter-part, which requires that (the bodies of) $\mathcal{M}$ and p do not have a common critical tuple; here, a *critical tuple* of a CQ q is a fact α such that there is an instance $\mathcal{D}$ with different sets of answers to q on $\mathcal{D}$ and $\mathcal{D} \setminus \{\alpha\}$, and a critical tuple of a CQ is such a tuple for every set containing this CQ (see [52] for more details on critical tuples).

A number of works focus not on policy analysis at design time, as we do, but on policy enforcement at query time. Calvanese et al. [24] study privacy-aware data access in the presence of ontologies, by extending the database authorisation framework by Zhang and Mendelzon [78]. In their setting, users are assigned a set of authorisation views, and each query is then answered by the system using only the information that follows from the ontology and their respective views. In contrast to our approach, the framework in [24, 78] does not provide means for explicitly representing sensitive information—answers to user queries that do not follow from the ontology and the materialisation of the views over the datasource are assumed to be sensitive by default; as a consequence, their technical results are incomparable to ours. From a practical perspective, our results can be seen as complementing the framework of [24, 78]: the compliance problems studied in our paper can be exploited at design time to validate a set of authorisation views against a policy, thus ensuring that the views cannot reveal sensitive information.

In the *controlled query evaluation* (*CQE*) framework, a *censor* ensures that query answers that may compromise the policy are either distorted, or not returned to users. CQE was introduced by [71] for databases and has received significant attention since (e.g., see [13, 14, 15]). CQE has been recently extended to ontologies in [16, 30, 31, 74]. Furthermore, [41] compares policy enforcement and policy restriction based approaches, in the absence of an ontology but for richer query languages (e.g., full relational calculus).

Chirkova and Yu [26] considered the information-leak disclosure problem, which is similar to our compliance. In our terms, the input to their problem is a set of CQ views $\mathcal{M}$, a CQ policy p, a set Σ of source constraints—that is, TGDs and *equality generating dependencies* (*EGDs*) over the source schema,—and an instance MV over the target schema. The question is whether there is a potential *information leak*—that is, a tuple $\mathbf{a}$ of constants such that $p(\mathbf{a})$ holds in every source instance $\mathcal{D}$ such that $\mathcal{D} \models \Sigma$ and $\mathcal{V}_{\mathcal{M},\mathcal{D}} = MV$. So, the difference between information leak disclosure and compliance (over CQ views) is that source constraints are allowed, ontology is disallowed, and the virtual image is given instead of a source instance. Chirkova and Yu [26] developed an algorithm that constructs all information leaks, provided Σ is weakly-acyclic [37]. They also showed that the algorithm works in polynomial space in the size of MV and p (i.e., under the assumption that $\mathcal{M}$ and Σ are fixed). In fact, an inspection of the algorithm shows that the assumption on $\mathcal{M}$ is not necessary, and the size of $\mathcal{M}$ may be counted without changing the complexity bound. We can exploit this algorithm to solve $\textsc{Comply}(\mathcal{O}, \mathcal{M}, \mathcal{D}, p)$ for the case when $\mathcal{O}$ is empty and $\mathcal{M}$ is a set of CQ views: we just need to materialise the virtual image $\mathcal{V}_{\mathcal{M},\mathcal{D}}$ and input it to the algorithm as MV, together with empty Σ as well as with unchanged $\mathcal{M}$ and p. However, the materialisation may be expensive in general, and to preserve the PSPACE complexity we should require bounded target arity (and hence bounded frontier)—this guarantees that the materialisation can be done in polynomial time. Nonetheless, our results in *Case (2)* of Theorem 19 and *Case (1)* of Theorem 25 show that the PSPACE complexity bound is not optimal, and this version of compliance is Π_3^p-complete. We also hypothesise that the ideas of the algorithm underlying Theorem 19 can be exploited to improve the PSPACE upper bound for the information-leak disclosure problem both for empty and for weakly-acyclic (but fixed) source constraints Σ.

Benedikt et al. [7] considered the scenario where the materialised contents of some relations over a relational schema are visible to users, whereas the contents of all other relations are invisible. A background theory provides semantic information

about both visible and invisible relations. The secret information is specified by a Boolean UCQ, and the goal is to determine whether (positive or negative) information about the UCQ can be derived by looking only at the contents of the visible relations and the background theory. A data-independent version is defined accordingly. Disclosure in this setting is related to the problem of query answering with "closed predicates", which has drawn much recent attention in the knowledge representation community [2, 61]. For CQ views, empty ontology, and Boolean policy, our both data-dependent and data-independent compliance problems can be reduced to the corresponding settings in [7]. In particular, given a set of CQ views $\mathcal{M}$, a source dataset $\mathcal{D}$, and a Boolean policy p as an input to COMPLY, we can create an input to the data-dependent disclosure problem of [7] as follows: let the target relational names of $\mathcal{M}$ be visible and the source relational names be invisible; then let, for each CQ view $\varphi(\mathbf{x}, \mathbf{z}) \rightarrow \mathsf{P}(\mathbf{x})$ in $\mathcal{M}$, the background theory contain the corresponding integrity constraint $\forall \mathbf{x}. (\exists \mathbf{z}. \varphi(\mathbf{x}, \mathbf{z}) \leftrightarrow \mathsf{P}(\mathbf{x}))$; finally, let the data instance be $\mathcal{D}$ and the secret query be p. It is easy to verify that COMPLY$(\emptyset, \mathcal{M}, \mathcal{D}, p)$ is `true` if and only if there is no disclosure for the constructed input. Moreover, since the reduction above is uniform in data, we get a corresponding reduction from COMPLYALL to the data-independent version of the disclosure problem of [7]. The first reduction can be used to establish decidability results for COMPLY, but the bounds that follow are not tight: the results of [7] only provide the 2EXPTIME upper bound in general and the EXPTIME upper bound in data complexity. The second, data-independent reduction is used, as Fact 41, in this paper in the proof of Theorem 44 on decidability of COMPLYALL for the case of CQ views and empty ontology. Benedikt et al. also provide an undecidability result for the general data-independent version of their problem, and the proof of our Theorem 30 exploits similar techniques. The undecidability result of [7], however, requires constraints involving the source relational names, which are not allowed in our setting; therefore, the result does not directly imply Theorem 30. Beyond GAV mappings, there is no obvious connection between our work and [7], because GLAV mappings coupled with an ontology over the global schema are incompatible with the assumptions of [7].

Finally, source indistinguishability is related to query inseparability in knowledge bases as studied by [18]. However, the emphasis in query inseparability is on having distinct ontologies (and not data) and mappings are not present; as a result, the techniques applied are different. Data-independent variants of ontology inseparability were studied in [11, 51].

10. Conclusions and Future Work

In this paper, we have provided an analysis of disclosure of source data in ontology-based integration. For indistinguishability, we have been able to prove tractability in data complexity for standard OBDA. In the case of compliance problems, we have still obtained very general decidability results; however, achieving tractability for standard OBDA requires additional restrictions on both the mappings and the policy. Finally, we have shown that instance-independent compliance is decidable for the important case of CQ view mappings and empty ontology, but becomes undecidable as soon as this setting is slightly generalised. As shown in Section 8, our results provide some significant modifications of the complexity landscape outlined in prior work.

Our results lead to a number of directions for future research. From a theoretical perspective, it would be interesting to address the problems mentioned next.

- The decidability (and complexity) of COMPLYALL for LAV mappings, which was left open in Section 7.
- The computational properties of STRCOMPLYALL, which remain completely unexplored.
- Databases are typically equipped with different types of integrity constraints such as primary and foreign keys, which may provide valuable additional information to an attacker. We believe that most of our decidability results extend to the setting where source constraints are allowed. It would be interesting to confirm this hypothesis and study the impact of source constraints on the complexity of our reasoning problems (by this, extending the results of Chirkova and Yu [26]).
- Our notion of compliance does not limit the computational resources of the attacker. Although Lemma 3 shows that the attacker can always make due with polynomially many queries, our lower bounds can be seen as showing that it may be hard in general for an attacker to determine if the policy holds. Thus, a main open issue is to distinguish the combinations of schemata and queries that are computationally easy for an attacker from those that are hard (as the data varies). Lutz et al. in [60] and [61] performed a similar kind of analysis for hybrid closed-and-open world query answering, and their techniques may be directly relevant.

From a parctical perspective, we see our complexity bounds for source indistinguishability and (strong) compliance in the setting of standard OBDA as rather favourable. In particular, source indistinguishability has AC^0 data complexity and, if we assume mappings to be LAV, it is also tractable in combined complexity; in turn, compliance and strong compliance both have AC^0 data complexity under the assumption that mappings are LAV and the policy is ground. We believe that these are all reasonable assumptions in practice, and we are planning to develop and implement practical, rewriting-based, compliance checking algorithms for the aforementioned cases. In contrast, our bounds for the compliance problem in more general settings (and especially the intractability results in data complexity), paint a rather pessimistic picture. We conjecture that for source databases arising in practice, the data can be abstracted—for example, by grouping values with similar types—in such a way that

the abstraction is orders of magnitude smaller than the source data, while compliance results about the source can be inferred from compliance results about the abstraction. We believe that criteria for a sound abstraction technique can be extracted from the critical instance technique described in Section 7. We will investigate this technique both from a practical and theoretical perspective in the future.

Acknowledgements

This research has been supported by the Royal Society under a University Research Fellowship, as well as by the EPSRC projects DBonto (EP/L012138/1), PDQ (EP/M005852/1), and ED3 (EP/N014359/1).

[1] Abiteboul, S., Hull, R., Vianu, V., 1995. Foundations of databases. Addison-Wesley.
[2] Ahmetaj, S., Ortiz, M., Simkus, M., 2016. Polynomial Datalog rewritings for expressive description logics with closed predicates. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2017). pp. 878–885.
[3] Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M., 2009. The DL-Lite family and relations. Journal of Artiicial Intelligence Research (JAIR) 36, 1–69.
[4] Baader, F., Bienvenu, M., Lutz, C., Wolter, F., 2016. Query and predicate emptiness in ontology-based data access. Journal of Artiicial Intelligence Research (JAIR) 56, 1–59.
[5] Bayardo Jr., R. J., Agrawal, R., 2005. Data privacy through optimal k-anonymization. In: Proceedings of the 21st International Conference on Data Engineering (ICDE 2005). pp. 217–228.
[6] Benedikt, M., Bourhis, P., ten Cate, B., Puppis, G., 2015. Querying visible and invisible tables in the presence of integrity constraints. CoRR abs/1509.01683.
[7] Benedikt, M., Bourhis, P., ten Cate, B., Puppis, G., 2016. Querying visible and invisible information. In: Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2016). pp. 297–306.
[8] Benedikt, M., Cuenca Grau, B., Kostylev, E. V., 2017. Source information disclosure in ontology-based data integration. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI 2017). pp. 1056–1062.
[9] Berger, R., 1966. The undecidability of the domino problem. Memoirs of the American Mathematical Society 66 (72).
[10] Bertossi, L. E., Li, L., 2013. Achieving data privacy through secrecy views and null-based virtual updates. IEEE Transactions on Knowledge and Data Engineering (TKDE) 25 (5), 987–1000.
[11] Bienvenu, M., Rosati, R., 2016. Query-based comparison of mappings in ontology-based data access. In: Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning (KR 2016). pp. 197–206.
[12] Bienvenu, M., ten Cate, B., Lutz, C., Wolter, F., 2014. Ontology-based data access: A study through disjunctive Datalog, CSP, and MMSNP. ACM Transactions on Database Systems (TODS) 39 (4), 33:1–33:44.
[13] Biskup, J., Bonatti, P., 2004. Controlled query evaluation for enforcing confidentiality in complete information systems. International Journal of Information Security (IJIS) 3 (1), 14–27.
[14] Biskup, J., Weibert, T., 2008. Keeping secrets in incomplete databases. International Journal of Information Security (IJIS) 7 (3), 199–217.
[15] Bonatti, P., Kraus, S., Subrahmanian, V. S., 1995. Foundations of secure deductive databases. IEEE Transactions on Data and Knowledge Engineering (TKDE) 7 (3), 406–422.
[16] Bonatti, P. A., Sauro, L., 2013. A confidentiality model for ontologies. In: Proceedings of the 12th International Semantic Web Conference (ISWC 2013), Part I. pp. 17–32.
[17] Börger, E., Grädel, E., Gurevich, Y., 1997. The classical decision problem. Perspectives in Mathematical Logic. Springer.
[18] Botoeva, E., Kontchakov, R., Ryzhikov, V., Wolter, F., Zakharyaschev, M., 2016. Games for query inseparability of description logic knowledge bases. Artificial Intelligence (AI) 234, 78–119.
[19] Calì, A., Gottlob, G., Lukasiewicz, T., 2012. A general Datalog-based framework for tractable query answering over ontologies. Journal of Web Semantics (JWS) 14, 57–83.
[20] Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G., 2017. Ontop: answering SPARQL queries over relational databases. Semantic Web (SW) 8 (3), 471–487.
[21] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rodriguez-Muro, M., Rosati, R., Ruzzi, M., Savo, D. F., 2011. The MASTRO system for ontology-based data access. Semantic Web (SW) 2 (1), 43–53.
[22] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R., 2005. DL-Lite: tractable description logics for ontologies. In: Proceedings of the 20th national conference on artificial intelligence and the 17th innovative applications of artificial intelligence conference (AAAI-IAAI 2005). pp. 602–607.
[23] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R., 2007. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. Journal of Automated Reasoning (JAR) 39 (3), 385–429.
[24] Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R., 2012. View-based query answering in description logics: Semantics and complexity. Journal of Computer and System Sciences (JCSS) 78 (1), 26–46.
[25] Calvanese, D., Dragone, L., Nardi, D., Rosati, R., Trisolini, S., 2006. Enterprise modeling and data warehousing in Telecom Italia. Information Systems (IS) 31 (1), 1–32.
[26] Chirkova, R., Yu, T., 2017. Exact detection of information leakage: Decidability and complexity. LNCS Theory Large-Scale Data- and Knowledge-Centered Systems (T-LSD-KCS) 32, 1–23.
[27] Cuenca Grau, B., Horrocks, I., 2008. Privacy-preserving query answering in logic-based information systems. In: Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008). pp. 40–44.
[28] Cuenca Grau, B., Horrocks, I., Krötzsch, M., Kupke, C., Magka, D., Motik, B., Wang, Z., 2013. Acyclicity notions for existential rules and their application to query answering in ontologies. Journal of Artificial Intelligence Research (JAIR) 47, 741–808.
[29] Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P. F., Sattler, U., 2008. OWL 2: The next step for OWL. Journal of Web Semantics (JWS) 6 (4), 309–322.
[30] Cuenca Grau, B., Kharlamov, E., Kostylev, E. V., Zheleznyakov, D., 2013. Controlled query evaluation over OWL 2 RL ontologies. In: Proceedings of the 12th International Semantic Web Conference (ISWC 2013), Part I. pp. 49–65.
[31] Cuenca Grau, B., Kharlamov, E., Kostylev, E. V., Zheleznyakov, D., 2015. Controlled query evaluation for Datalog and OWL 2 profile ontologies. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015). pp. 2883–2889.

[32] Cuenca Grau, B., Kostylev, E. V., 2016. Logical foundations of privacy-preserving publishing of Linked Data. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016). pp. 943–949.
[33] Deep, S., Koutris, P., 2017. The design of arbitrage-free data pricing schemes. In: Proceedings of the 20th International Conference on Database Theory (ICDT 2017). pp. 12:1–12:18.
[34] Dix, J., Faber, W., Subrahmanian, V. S., 2005. The relationship between reasoning about privacy and default logics. In: Proceedings of the 12th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2005). pp. 637–650.
[35] Doan, A., Halevy, A. Y., Ives, Z. G., 2012. Principles of Data Integration. Morgan Kaufmann.
[36] Dwork, C., 2006. Differential privacy. In: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP 2006), Part II. pp. 1–12.
[37] Fagin, R., Kolaitis, P. G., Miller, R. J., Popa, L., 2005. Data exchange: semantics and query answering. Theoretical Computer Science (TCS) 336 (1), 89–124.
[38] Giese, M., Soylu, A., Vega-Gorgojo, G., Waaler, A., Haase, P., Jiménez-Ruiz, E., Lanti, D., Rezk, M., Xiao, G., Özçep, Ö. L., Rosati, R., 2015. Optique: Zooming in on big data. IEEE Computer 48 (3), 60–67.
[39] Gogacz, T., Marcinkowski, J., 2014. All-instances termination of chase is undecidable. In: Proceedings of the 41st International Colloquium on Automata, Languages and Programming (ICALP 2014), Part II. pp. 293–304.
[40] Gottlob, G., Kikot, S., Kontchakov, R., Podolskii, V. V., Schwentick, T., Zakharyaschev, M., 2014. The price of query rewriting in ontology-based data access. Artificial Intelligence (AI) 213, 42–59.
[41] Guarnieri, M., Basin, D. A., 2014. Optimal security-aware query processing. Proceedings of the VLDB Endowment (PVLDB) 7 (12), 1307–1318.
[42] Gurevich, Y. S., Koryakov, I. O., 1972. Remarks on Berger's paper on the domino problem. Siberian Mathematical Journal 13 (2), 319–321.
[43] Halevy, A. Y., 2001. Answering queries using views: A survey. VLDB Journal 10 (4), 270–294.
[44] Horrocks, I., Giese, M., Kharlamov, E., Waaler, A., 2016. Using semantic technology to tame the data variety challenge. IEEE Internet Computing 20 (6), 62–66.
[45] Hull, R., 1997. Managing semantic heterogeneity in databases: A theoretical perspective. In: Proceedings of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1997). pp. 51–61.
[46] Jiménez-Ruiz, E., Kharlamov, E., Zheleznyakov, D., Horrocks, I., Pinkel, C., Skjæveland, M. G., Thorstensen, E., Mora, J., 2015. BootOX: practical mapping of RDBs to OWL 2. In: Proceedings of the 14th International Semantic Web Conference (ISWC 2015), Part II. pp. 113–132.
[47] Johnson, D. S., Klug, A. C., 1984. Testing containment of conjunctive queries under functional and inclusion dependencies. Journal of Computer and System Sciences (JCSS) 28 (1), 167–189.
[48] Kharlamov, E., Brandt, S., Jiménez-Ruiz, E., Kotidis, Y., Lamparter, S., Mailis, T., Neuenstadt, C., Özçep, Ö. L., Pinkel, C., Svingos, C., Zheleznyakov, D., Horrocks, I., Ioannidis, Y. E., Möller, R., 2016. Ontology-based integration of streaming and static relational data with Optique. In: Proceedings of the 2016 International Conference on Management of Data, (SIGMOD 2016). pp. 2109–2112.
[49] Kharlamov, E., Hovland, D., Jiménez-Ruiz, E., Lanti, D., Lie, H., Pinkel, C., Rezk, M., Skjæveland, M. G., Thorstensen, E., Xiao, G., Zheleznyakov, D., Horrocks, I., 2015. Ontology based access to exploration data at Statoil. In: Proceedings of the 14th International Semantic Web Conference (ISWC 2015), Part II. pp. 93–112.
[50] Kikot, S., Kontchakov, R., Podolskii, V. V., Zakharyaschev, M., 2014. On the succinctness of query rewriting over shallow ontologies. In: Proceedings of the Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic and the 28th Annual ACM/IEEE Symposium on Logic in Computer Science (CSL-LICS 2014). pp. 57:1–57:10.
[51] Konev, B., Kontchakov, R., Ludwig, M., Schneider, T., Wolter, F., Zakharyaschev, M., 2011. Conjunctive query inseparability of OWL 2 QL TBoxes. In: Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI 2011). pp. 221–226.
[52] Kostylev, E., Suciu, D., 2018. A note on the hardness of the critical tuple problem. CoRR abs/1804.00443.
[53] Koutris, P., Upadhyaya, P., Balazinska, M., Howe, B., Suciu, D., 2013. Toward practical query pricing with QueryMarket. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2013). pp. 613–624.
[54] Koutris, P., Upadhyaya, P., Balazinska, M., Howe, B., Suciu, D., 2015. Query-based data pricing. Journal of the ACM 62 (5).
[55] Lembo, D., Mora, J., Rosati, R., Savo, D. F., Thorstensen, E., 2014. Towards mapping analysis in ontology-based data access. In: Proceedings of the 8th International Conference on Web Reasoning and Rule Systems (RR 2014). pp. 108–123.
[56] Lenzerini, M., 2002. Data integration: A theoretical perspective. In: Proceedings of the 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2012). pp. 233–246.
[57] Li, C., Li, D. Y., Miklau, G., Suciu, D., 2014. A theory of pricing private data. ACM Transactions on Database Systems (TODS) 39 (4), 34:1–34:28.
[58] Li, C., Miklau, G., 2012. Pricing aggregate queries in a data marketplace. In: Proceedings of the 15th International Workshop on the Web and Databases (WebDB 2012). pp. 19–24.
[59] Lin, B., Kifer, D., 2014. On arbitrage-free pricing for general data queries. Proceedings of the VLDB Endowment (PVLDB) 7 (9), 757–768.
[60] Lutz, C., Seylan, I., Wolter, F., 2012. Mixing open and closed world assumption in ontology-based data access: Non-uniform data complexity. In: Proceedings of the 25th International Workshop on Description Logics (DL 2012). pp. 268–278.
[61] Lutz, C., Seylan, I., Wolter, F., 2015. Ontology-mediated queries with closed predicates. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015). pp. 3120–3126.
[62] Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M., 2007. *L*-diversity: Privacy beyond *k*-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD) 1 (1).
[63] Marnette, B., 2010. Tractable schema mappings under oblivious termination. Ph.D. thesis, University of Oxford, UK.
[64] Miklau, G., Suciu, D., 2007. A formal analysis of information disclosure in data exchange. Journal of Computer and System Sciences (JCSS) 73 (3), 507–534.
[65] Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C., 2012. OWL 2 Web ontology language profiles (2nd edition). W3C Recommendation.
[66] Nash, A., Deutsch, A., 2007. Privacy in GLAV information integration. In: Proceedings of the 11th International Conference on Database Theory (ICDT 2007). pp. 89–103.
[67] Papadimitriou, C. H., 1994. Computational complexity. Addison-Wesley.
[68] Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R., 2008. Linking data to ontologies. LNCS Journal on Data Semantics (JDS) 10, 133–173.
[69] Samarati, P., 2001. Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering (TKDE) 13 (6), 1010–1027.
[70] Shmueli, O., 1993. Equivalence of Datalog queries is undecidable. Journal of Logic Programming (JLP) 15 (3), 231–241.
[71] Sicherman, G. L., de Jonge, W., van de Riet, R. P., 1983. Answering queries without revealing secrets. ACM Transactions on Database Systems (TODS) 8 (1), 41–59.
[72] Spakowski, H., 2005. Completeness for parallel access to NP and counting class separations. Ph.D. thesis, Universität Düsseldorf.

[73] Stouppa, P., Studer, T., 2006. A formal model of data privacy. In: Proceedings of the 6th International Andrei Ershov Memorial Conference on Perspectives of Systems Informatics (PSI 2006). pp. 400–408.
[74] Studer, T., Werner, J., 2014. Censors for Boolean description logic. Transactions on Data Privacy (TDP) 7 (3), 223–252.
[75] Sweeney, L., 2002. k-Anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10 (5), 557–570.
[76] Ullman, J. D., 1997. Information integration using logical views. In: Proceedings of the 6th International Conference on Database Theory (ICDT 1997). pp. 19–40.
[77] Wagner, K. W., 1987. More complicated questions about maxima and minima, and some closures of NP. Theoretical Computer Science (TCS) 51, 53–80.
[78] Zhang, Z., Mendelzon, A. O., 2005. Authorization views and conditional query containment. In: Proceedings of the 10th International Conference on Database Theory (ICDT 2005). pp. 259–273.