Query Answering over Complete Data with Conceptual Constraints

Nhung Ngo
Supervisor: Enrico Franconi
Expected graduation date: in February 2017
Free University of Bolzano, Italy
ngo@inf.unibz.it

ABSTRACT

Query answering over databases with conceptual constraints is an important problem in database theory. To deal with the problem, the ontology-based data access approach uses ontologies to capture both constraints and databases. In this approach, databases are considered under open-world assumption which creates many issues including the necessity of restricting to only positive queries, and the failure of query composition. In our research, we focus on a combined approach that allows data in databases stays completely as under closed-world assumption while knowledge providing by conceptual constraints can be incomplete. We first study the complexity of query answering problem under description logic constraints in the presence of complete data and show that complete data makes query answering become harder than query answering over incomplete data only. We then provide a query rewriting technique that supports deciding the existence of a safe-range first-order equivalent reformulation of a query in terms of the database schema, and if so, it provides an effective approach to construct the reformulation. Since the reformulation is a safe-range formula, it is effectively executable as an SQL query. At the end, we study the definability abduction problem which aims to characterize the least committing extensions of conceptual constraints to gain the exact rewritable of queries. We also apply this idea to data exchange - where we want to characterize the case of lossless transformations of data.

Keywords

database; complete data; data exchange

1. INTRODUCTION

This thesis addresses the problem of query answering with expressive constraints over complete data in which complete data is stored in a classical finite relational database and constraints provide additional knowledge and conceptual views of the database. The vocabulary of constraints extends the basic vocabulary of the database. Querying a database using the terms in such a richer language allows for more flexibility than using only the basic vocabulary of the relational database directly.

In the literature, complete data stored in a database has been called the *closed predicates* [22, 26], *exact views* [23, 24], or *DBox* [32]. In our research, we use DBox to refer to complete data. Basically, a DBox is a set of ground atoms which semantically behaves like a database, i.e., the interpretation of the database predicates in the DBox is exactly equal to the database relations. The DBox predicates are *closed*, i.e., their extensions are the same in every interpretation. We do not consider here the *open* interpretation for complete data (also called *ABox* or *sound views*). In an ABox, an interpretation of database predicates contains the database relations and possibly more. Therefore, the notion of ABox is less faithful in the representation of a database semantics since it would allow for spurious interpretations of database predicates with additional unwanted tuples not present in the original database. As an example, consider a ground negative query over a given standard relational database. By adding an ontology on top of it, its answer is not supposed to change since the query uses only the signature of the database and additional constraints are not supposed to change the meaning of the query. Whereas if the database were treated as an ABox (sound views) the answer may change in presence of an ontology.

In contrast, constraints can provide incomplete knowledge. In other words, non-DBox predicates in the constraints are *open*, i.e., their extensions may vary among different interpretations. For example, a municipality-provided table of bus routes can be assumed complete while the set of bus drivers can be incomplete. To capture incomplete knowledge, description logic (DL) ontologies are often used. Therefore, it is necessary to consider the case of query answering with constraints written in some description logics.

To summarize, our query answering setting is a tuple $\mathcal{M} = (\Sigma, \mathcal{K}, \mathcal{D}, q(\vec{x}))$ such that:

- Signature Σ is the set of first-order predicates.
- Set of contraints $\mathcal{K}$ are written using predicates in Σ in some fragment of first-order logic such as tuple-generating-dependencies (tgds) or description logics.
- DBox $\mathcal{D}$ contains the DBox predicates and its interpretation.
- Query $q(\vec{x})$ is a first-order logical formula.

Based on the setting, the thesis studies the following research questions.

1. Combined complexity of query answering with com-

SIGMOD'16 PhD Symp, June 25-July 01 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2926693.2929899

plete data: what is the computational complexity of checking if a tuple $\vec{a}$ is in the certain answer of $q(\vec{x})$ under $\mathcal{K} \cup \mathcal{D}$, i.e for every model $\mathcal{I}$ of $\mathcal{K} \cup \mathcal{D}$, $\mathcal{I} \models q(\vec{a})$?

Unlike previous works where only data complexity was studied [9, 22] , we consider here the *combined complexity*, that is, not only the data is considered as an input, but also the constraints, and the query.

2. Exact query reformulation: can $q(\vec{x})$ be rewritable in terms of DBox predicates? If it is the case, how to compute the rewriting and how to reduce the query answering problem over $\mathcal{K} \cup \mathcal{D}$ to a relational algebra evaluation over only $\mathcal{D}$? The query reformulation problem has received strong interest in classical relational database research as well as modern knowledge representation studies. The mainstream research on query reformulation [16] mostly is based on perfect or maximally contained rewritings with sound views under relatively inexpressive constraints [1]. Different to the approach, we focus here on exact rewritings, since it characterizes precisely the query answering problem with constraints and complete data, in the case when the exact semantics of the complete data must be preserved.

3. Definability abduction: given a query answering setting $\mathcal{M}$ in which $q(\vec{x})$ is not rewritable over DBox predicates under $\mathcal{K}$, how to characterize a least committing extension Δ of $\mathcal{K}$ such that $q(\vec{x})$ is rewritable over DBox predicates under $\mathcal{K} \cup \Delta$? To the best of our knowledge, this is the first work in which abductive reasoning is applied to extend a conceptual schema according to the requirement of exact query reformulation. We also study an interesting application in data exchange in which we believe it is necessary to have such extensions.

2. COMBINED COMPLEXITY ANALYSIS

In this section, we summarize our results in combined complexity analysis of query answering with complete data (please refer to [26] for details). We focus on the case where constraints are written using DL fragments and positive queries. The main query language we consider is *Boolean conjunctive queries (BCQs)*, which take the form $q = \alpha_1 \wedge \ldots \wedge \alpha_n$, where each α_j is an atom of the form $A(t)$ or $r(t, t')$ with A is a concept name, r is a role name, and $\{t, t'\}$ are constants or individuals. A *Boolean union of conjunctive queries (BUCQs)* is of the form $q' = q_1 \vee \ldots \vee q_n$, where each q_i is a BCQ. In case $q(\vec{x})$ is such that each variable in q also occurs in $\vec{x}$, we call it *quantifier-free* (a *qfUCQ*). A qfUCQ with just one atom is called an *instance query*. Given $\mathcal{K}$, $q(\vec{x})$ and $\vec{a}$ as above, the *query answering problem* is to decide $\mathcal{K} \models q(\vec{a})$.

A summary of our result and its comparison to the case without DBox are provided in Table 1.

Despite the above negative results, we can still identify some useful classes of queries with lower complexity. In particular, we considered a restriction on variables called *$\mathcal{K}$-safety* which guarantees that it is sufficient to consider assignments of the variable to individuals in the KB, and there is no need to consider other objects in the interpretation domain . A query is called *$\mathcal{K}$-safe* if all its variables are $\mathcal{K}$-safe. We also generalize $\mathcal{K}$-safe queries to *$\mathcal{K}$-acyclic* by allowing for variables that are not $\mathcal{K}$-safe, but requiring that they induce only *acyclic* subqueries in the original query. Please refer to Table 2 for the complexity results of answering these queries.

3. EXACT QUERY REFORMULATION

In this section, we explain briefly our exact query reformulation framework where constraints and queries are expressed in first-order logic. Related results are published in [11, 10].

3.1 Definable queries

The certain answer of a query includes all the substitutions which make the query true in *all* the models of the constraints. In other words, it may be the case that the answer to the query is not necessarily the same among all the models of the constraints. In this case, the query is not fully determined by the given source data; and there is some answer which is possible, but not certain. Therefore, we focus on the case when a query has the same answer over all the models of the constraints, namely, on the case when the information requested by the query is fully available from the DBox without ambiguity.

In fact, the *determinacy* of a source database with respect to a query [24, 23] is identical to the *implicit definability* of a formula (the query) from a set of predicates (the database predicates)[5].

Definition 3.1 (**Implicit Definability**). *Let $\mathcal{I}$ and $\mathcal{J}$ be any two models of $\mathcal{K} \cup \mathcal{D}$. Query $q(\vec{x})$ is implicitly definable from the DBox predicates under $\mathcal{K}$ iff for every tuple $\vec{a}$ $\mathcal{I} \models q(\vec{a})$ if and only if $\mathcal{J} \models q(\vec{a})$.*

A query is implicitly definable if its truth value in any model of the constraints depends *only* on the domain, on the interpretation of the DBox predicates, and on the interpretation of the constants. The answer of an implicitly definable query does not depend on the interpretation of non-DBox predicates. Once the DBox and a domain are fixed, it is never the case that a substitution would make the query true in some model of the constraints and false in other.

The *exact reformulation* of a query [24] (also called in the logic literature *explicit definition* [5]) is a formula logically equivalent to the query which makes use *only* of database predicates and constants.

Definition 3.2 (**Exact Reformulation**). *Query $q(\vec{x})$ is explicitly definable from the DBox predicates under constraints $\mathcal{K}$ iff there is some formula $\widehat{q}(\vec{x})$, such that $\mathcal{K} \models \forall \vec{x}. q(\vec{x}) \leftrightarrow \widehat{q}(\vec{x})$ and $\widehat{q}(\vec{x})$ is written using DBox predicates only. We call this formula $\widehat{q}(\vec{x})$ an exact reformulation of $q(\vec{x})$ under $\mathcal{K}$ over $\mathcal{D}$.*

Definability of a query is completely characterised by the existence of an exact reformulation of the query based on the following theorem:

Theorem 1 (**Projective Beth theorem** [5]). *Query $q(\vec{x})$ is implicitly definable from the DBox predicates under contraints $\mathcal{K}$, iff it is explicitly definable as a formula $\widehat{q}(\vec{x})$ written using DBox predicates under $\mathcal{K}$.*

Let $\mathcal{Q}$ be any formula and $\widetilde{\mathcal{Q}}$ the formula obtained from it by uniformly replacing every occurrence of each non-DBox

	Without DBox		With DBox	
	Instance query answering	B(U)CQs answering	Instance query answering	B(U)CQs answering
$\mathcal{EL}$	P [2]	NP [18, 30]	Exp	2Exp
$\textit{DL-Lite}_{core}$	NL [1]	NP [1]	NP	$\geq$ coNExp
$\textit{DL-Lite}_{\mathcal{R}}$, $\textit{DL-Lite}^{enum}_{\mathcal{R},bool}$	P [1]	NP [1]	NP	$\geq$ 2Exp
$\mathcal{ELIF}$ Horn-$\mathcal{SHIQ}$	Exp [20]	Exp [7]	NExp	$\geq$ N2Exp [15] decidable [31] / $\leq$ open*
$\mathcal{ELOIF}$, Horn-$\mathcal{SHOIQ}$	Exp [27]	Exp [28]	NExp	$\geq$ N2Exp [15] decidable [31] / $\leq$ open*
$\mathcal{ALCO}$	Exp [34, 13]	2Exp	Exp	2Exp
$\mathcal{SHOQ}, \mathcal{SHOI}$	Exp [34, 17, 13]	2Exp [6, 14]	Exp	2Exp
$\mathcal{SHOIQ}$	NExp [33]	$\geq$ N2Exp [15] $\leq$ open*	NExp	$\geq$ N2Exp [15] $\leq$ open*

Table 1: Combined complexity of reasoning in description logics with/without closed predicates. By $\geq$ we indicate lower bounds, by $\leq$ upper bounds, and the rest are all completeness results. For the cells marked with *, decidability if only simple roles occur in the query follows from [31], but no complexity upper bounds are known.

	$\mathcal{K}$-safe	$\mathcal{K}$-acyclic	every binary atom has a $\mathcal{K}$-safe variable
$\textit{DL-Lite}_{core}$	Π^P_2	in Exp	Π^P_2
$\textit{DL-Lite}_{\mathcal{R}}$	Π^P_2	in Exp	Π^P_2
$\mathcal{EL}$	Exp	Exp	coNExp-hard
$\mathcal{ALCO}(\mathcal{I})$	Exp	Exp	coNExp-hard

Table 2: Complexity of query answering with closed predicates (results are completeness unless otherwise stated).

predicate P with a new predicate $\widetilde{P}$. We extend this renaming operator $\widetilde{\cdot}$ to any set of formulas in a similar way. One can check whether a query is implicitly definable by using the following theorem.

Theorem 2 (**Testing Determinacy** [5]). *Query $q(\vec{x})$ is implicitly definable from the DBox predicates under the constraints $\mathcal{K}$ iff $\mathcal{K} \cup \widetilde{\mathcal{K}} \models \forall \vec{x}. q(\vec{x}) \leftrightarrow \widetilde{q}(\vec{x})$.*

3.2 Safe range rewritings

Given a query answering setting $\mathcal{M} = (\Sigma, \mathcal{K}, \mathcal{D}, q(\vec{x}))$ in which $q(\vec{x})$ is implicitly definable from the DBox predicates under $\mathcal{K}$, our ultimate goal is to find a safe-range reformulation $\widehat{q}(\vec{x})$ of $q(\vec{x})$, that being evaluated as a relational algebra expression over a legal database instance (e.g., using a relational database system with SQL) gives the same answer as the certain answer of $q(\vec{x})$ to the DBox under $\mathcal{K}$. This can be reformulated as the problem of finding an exact reformulation $\widehat{q}(\vec{x})$ of $q(\vec{x})$ under $\mathcal{K}$ as a safe-range query over DBox $\mathcal{D}$.

Since an exact reformulation is equivalent under the constraints to the original query, the certain answer of the original query and of the reformulated query are identical. The following theorem states the condition to reduce the original query answering problem – based on entailment – to the problem of checking the validity of the exact reformulation over a *single* model: the condition is that the reformulation should be domain independent.

Theorem 3. *Given a query answering setting $\mathcal{M}$, if $\widehat{q}(\vec{x})$ is an exact domain independent (or safe-range) reformulation of $q(\vec{x})$ under $\mathcal{K}$ over $\mathcal{D}$, then certain answer of $q(\vec{x})$ over $\mathcal{K} \cup D$ coincides with the answer of $\widehat{q}(\vec{x})$ over $\mathcal{D}$ under the domain containing all the constants in $\widehat{q}(\vec{x})$ and $\mathcal{D}$.*

A safe-range reformulation is *necessary* to transform a first-order query to a relational algebra query which can then be evaluated by using SQL techniques. The theorem above shows in addition that being safe-range is also a *sufficient* property for an exact reformulation to be correctly evaluated as an SQL query.

3.3 Finitely controllability

In order to be complete, our framework is applicable to constraints and queries expressed in any fragment of first-order logic enjoying finitely controllable determinacy [24], a stronger property than the finite model property of the logic. If the employed logic does not enjoy finitely controllable determinacy our approach would become sound but incomplete, but still effectively implementable using standard theorem proving techniques. We explore non-trivial applications where the framework is complete. For example, in the application with constraints written in $\mathcal{ALCHOI}$ and concept queries, we show how (i) to check whether the answers to a given query with a set of constraints are *solely* determined by the extension of the DBox predicates and, if so, (ii) to find an equivalent rewriting of the query in terms of the DBox predicates to allow the use of standard database technology for answering the query. This means we benefit from the low computational complexity in the size of the data for answering queries on relational databases.

4. DEFINABILITY ABDUCTION

In this section, we first formalize the problem and a semantic criteria to compare between its solution in the general setting and then summarize its application in gaining unique solution in data exchange. For details, please refer to[12, 25].

4.1 General problem

Based on Theorem 2, we know that testing definability can be done by logical entailment checking. Therefore, to answer the research question of fixing constraints to gain definability, we use an approach similar to abductive reasoning [29] and introduce the following general problem. Here $\sigma(\phi)$ means the signature of ϕ where ϕ is some set of formulas/tuples.

Definition 4.1. *Let* $\mathcal{K}$ *be a set of first order sentences. A triple* $D = (\mathcal{P}, p, \mathcal{K})$ *is a* definability abductive setting *(DAS) if it holds that* $\mathcal{K} \cup \widetilde{\mathcal{K}} \not\models \forall \bar{x}.p(\bar{x}) \leftrightarrow \tilde{p}(\bar{x})$*, where* $\tilde{p}$ *is a fresh predicate of the same arity as p, and* $\widetilde{\mathcal{K}}$ *is obtained from* $\mathcal{K}$ *by replacing all predicates from* $\sigma(\mathcal{K}) \setminus \mathcal{P}$ *with fresh predicates with the same arity.*

The non-entailment in Definition 4.1 means that predicate $p(\vec{x})$ is not definable from $\mathcal{P}$ under $\mathcal{K}$. A set of sentences Δ is called a *d-extension* to a DAS $D = (\mathcal{P}, p, \mathcal{K})$ if p is definable from $\mathcal{P}$ under $\mathcal{K} \cup \Delta$. A d-extension Δ is *minimal* if for every d-extension Δ_1 such that $\mathcal{K} \cup \Delta \models \Delta_1$ it holds that $\mathcal{K} \cup \Delta_1 \models \Delta$. These restrictions are natural as we are interested in finding meaningful d-extensions which minimally change the intended meaning of the constraints.

Given a query answering setting $\mathcal{M}$ in which $q(\vec{x})$ is not definable from $\mathcal{D}$ under $\mathcal{K}$, one can formalize a DAS $D_{\mathcal{M}} = (\sigma(\mathcal{D}), q, \mathcal{K})$ such that its minimal d-extension Δ is a possible least committing extension of $\mathcal{K}$ that provides the definability for $q(\vec{x})$.

4.2 Application in Data Exchange

The problem of data exchange was formally defined in [8] as the problem of transforming data structured under a source schema into data structured under a target schema. In data exchange, schema mappings written in the language of source-to-target tuple-generating-dependencies (s-t tgds) [8] and therefore, given a source instance, there are possibly many valid target instances. That is, the target database is actually an *incomplete database.* As a matter of fact, the problem of query answering over the target data is inherently complex and non-intuitive for general (non-positive) relational or aggregate queries. Intuitively, it is basically comparable to entailment with open-world semantics (i.e the computation of *certain answers*), and therefore standard relational database technologies can not be used. In other words, certain answer semantics gives no reasonable answers to non-monotone queries, as nicely summarised by [21]: answer to negative queries may be incomplete, and answering queries with aggregation may become trivial and non-informative.

To allow for general relational and aggregate queries, we enrich the data exchange framework, by suggesting "reasonable" amendments to the initial mapping. The newly obtained schema mapping will then produce a *unique* materialised target instance depending only on the given source database instance and the schema mapping. Consequently, we do not have anymore an incomplete database in the target schema. This can be done by considering a set of DAS setting $D_p = (\mathbf{S}, p, \mathcal{K})$ for each target predicate p, source schema $\mathbf{S}$ and schema mapping $\mathcal{K}$.

Given these DASs, we provide an algorithm to generate their minimal d-extensions as in the following example.

Example 4.2. *Consider the DAS* $(\mathbf{S}, Staff, \mathcal{K})$ *where* $\mathbf{S} = \{Employee, Manager\}$*, and* $\mathcal{K}$ *is the set of following full tgds:*
$Employee(x) \rightarrow Staff(x)$
$Manager(x) \rightarrow Staff(x)$
Then $\Delta = \{Staff(x) \rightarrow (Employee(x) \vee Manager(x))\}$ *is a minimal d-extension of* $(\mathbf{S}, Staff(x), \mathcal{K})$.

Besides, we also study the complexity of the d-extension checking problem in different settings. In the general setting we have undecidability results; once we restrict the syntax of input dependencies to be weakly guarded, the problem is 2EXPTIME-complete.

5. CONCLUSION

DBoxes reflect complete data, but as our results show, they make query answering computationally costly. It remains a challenge to identify useful restricted settings with lower complexity, and to explore other ways for effectively supporting partial completeness.

In terms of query reformulation with DBox, we have introduced a framework to compute the exact reformulation of first-order queries to a DBox under first-order constraints. We have found the exact conditions which guarantee that a safe-range reformulation exists, and we show that it can be evaluated as a relational algebra query over the database to give the same answer as the original query under the constraints. For future work, we are working on extending the theoretical framework with conjunctive queries: we need finitely controllable determinacy with conjunctive queries, which seems to follow for some description logic from the works by [4].

By definability abduction, we have considered the problem of gaining definability of target predicates over source predicates in data exchange. We have provided algorithms to generate d-extensions and have studied the complexity of the d-extension checking problem. In the future, we are interested in characterizing a semantic order among d-extensions to specify the "best" extension.

Acknowledgement

This research is partially supported by Free University of Bolzano, under project ORMiE.

6. REFERENCES

[1] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The dl-lite family and relations. *J. Artif. Int. Res.*, 36(1):1–69, Sept. 2009.

[2] F. Baader, S. Brand, and C. Lutz. Pushing the EL envelope. In *Proc. of IJCAI 2005*, pages 364–369. Morgan-Kaufmann Publishers, 2005.

[3] F. Baader, S. Brandt, and C. Lutz. Pushing the EL envelope further. In K. Clark and P. F. Patel-Schneider, editors, *In Proceedings of the OWLED 2008 DC Workshop on OWL: Experiences and Directions*, 2008.

[4] V. Bárány, G. Gottlob, and M. Otto. Querying the guarded fragment. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science (LICS 2010)*, pages 1–10, 2010.

[5] E. Beth. On Padoa's method in the theory of definition. *Indagationes Mathematicae*, 15:330–339, 1953.

[6] D. Calvanese, T. Eiter, and M. Ortiz. Regular path queries in expressive description logics with nominals. In *Proc. of IJCAI 2009*, pages 714–720, 2009.

[7] T. Eiter, G. Gottlob, M. Ortiz, and M. Šimkus. Query answering in the description logic Horn-$\mathcal{SHIQ}$. pages 166–179, Berlin, Heidelberg, 2008. Springer-Verlag.

[8] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[9] E. Franconi, Y. A. Ibáñez-García, and I. Seylan. Query answering with DBoxes is hard. *Electr. Notes Theor. Comput. Sci.*, 278:71–84, 2011.

[10] E. Franconi, V. Kerhet, and N. Ngo. Exact query reformulation with first-order ontologies and databases. In *Logics in Artificial Intelligence - 13th European Conference, JELIA 2012*, pages 202–214, 2012.

[11] E. Franconi, V. Kerhet, and N. Ngo. Exact query reformulation over databases with first-order and description logics ontologies. *Journal of Artificial Intelligence Research (JAIR)*, 48:885–922, 2013.

[12] E. Franconi, N. Ngo, and E. Sherkhonov. The definability abduction problem for data exchange - (extended abstract). In *Web Reasoning and Rule Systems - 6th International Conference, RR 2012*, pages 217–220, 2012.

[13] B. Glimm, I. Horrocks, and U. Sattler. Deciding shoqˆcap knowledge base consistency using alternating automata. In *Description Logics*, volume 353 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[14] B. Glimm, I. Horrocks, and U. Sattler. Unions of conjunctive queries in SHOQ. In *Proc. of the 11th International Conference on the Principles of Knowledge Representation and Reasoning (KR-08)*, pages 252–262. AAAI Press/The MIT Press, 2008.

[15] B. Glimm, Y. Kazakov, and C. Lutz. Status QIO: An update. volume 745 of *CEUR Workshop Proceedings*, 2011.

[16] A. Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10:270–294, December 2001.

[17] J. Hladik. A tableau system for the description logic SHIO. In *IJCAR Doctoral Programme*, 2004.

[18] M. Krötzsch and S. Rudolph. Conjunctive queries for $\mathcal{EL}$ with composition of roles. volume 250 of *CEUR Electronic Workshop Proceedings,* http://ceur-ws.org/Vol-250/, 2007.

[19] M. Krötzsch, S. Rudolph, and P. Hitzler. Conjunctive queries for a tractable fragment of OWL 1.1. In *The Semantic Web, 6th Int. Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 310–323. Springer, 2007.

[20] M. Krötzsch, S. Rudolph, and P. Hitzler. Complexities of Horn description logics. *ACM Trans. Comp. Log.*, 14(1):2:1–2:36, 2013.

[21] L. Libkin and C. Sirangelo. Open and closed world assumptions in data exchange. In *Description Logics 2009*, 2009.

[22] C. Lutz, I. Seylan, and F. Wolter. Ontology-based data access with closed predicates is inherently intractable(sometimes). In *Proc. of IJCAI 2013.* IJCAI/AAAI, 2013.

[23] M. Marx. Queries determined by views: pack your views. In *Proceedings of the 26th ACM symposium on Principles of Database Systems*, PODS '07, pages 23–30, 2007.

[24] A. Nash, L. Segoufin, and V. Vianu. Views and queries: Determinacy and rewriting. *ACM Trans. Database Syst.*, 35:21:1–21:41, July 2010.

[25] N. Ngo and E. Franconi. Unique solutions in data exchange. In *Database and Expert Systems Applications - 25th International Conference, DEXA 2014, Munich, Germany, September 1-4, 2014. Proceedings, Part II*, pages 281–294, 2014.

[26] N. Ngo, M. Ortiz, and M. Simkus. Closed predicates in description logics: Results on combined complexity. In *KR*. AAAI Press, April 2016.

[27] M. Ortiz, S. Rudolph, and M. Simkus. Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In *KR*, 2010.

[28] M. Ortiz, S. Rudolph, and M. Šimkus. Query answering in the Horn fragments of the description logics SHOIQ and SROIQ. In *Proc. of IJCAI 2011*, pages 1039–1044, 2011.

[29] G. Paul. Approaches to abductive reasoning: An overview. *AI Review*, 7:109–152, 1993.

[30] R. Rosati. On conjunctive query answering in $\mathcal{EL}$. volume 250 of *CEUR Electronic Workshop Proceedings,* http://ceur-ws.org/Vol-250/, 2007.

[31] S. Rudolph and B. Glimm. Nominals, inverses, counting, and conjunctive queries or: Why infinity is your friend! *J. Artif. Intell. Res. (JAIR)*, 39:429–481, 2010.

[32] İnanç. Seylan, E. Franconi, and J. de Bruijn. Effective query rewriting with ontologies over DBoxes. In *Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pages 923–925, 2009.

[33] S. Tobies. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *J. Artif. Int. Res.*, 12(1):199–217, May 2000.

[34] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation.* PhD thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany, 2001.