Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Improving Data Management Using Domain Knowledge

Magdalena Ortiz
Faculty of Informatics, TU Wien, Austria
ortiz@kr.tuwien.ac.at

Abstract

The development of tools and techniques for flexible and reliable data management is a long-standing challenge, ever more pressing in today's data-rich world. We advocate using domain knowledge expressed in ontologies to tackle it, and summarize some research efforts to this aim that follow two directions. First, we consider the problem of ontology-mediated query answering (OMQA), where queries in a standard database query language are enriched with an ontology expressing background knowledge about the domain of interest, used to retrieve more complete answers when querying incomplete data. We discuss some of our contributions to OMQA, focusing on (i) expressive languages for OMQA, with emphasis on combining the open- and closed-world assumptions to reason about partially complete data; and (ii) OMQA algorithms based on rewriting techniques. The second direction we discuss proposes to use ontologies to manage evolving data. In particular, we use ontologies to model and reason about constraints on datasets, effects of operations that modify data, and the integrity of the data as it evolves.

1 Introduction

The quest for flexible and reliable tools to manage large amounts of data and to extract useful information has been at the core of computer science research for decades. Far from over, the soaring amounts of data available nowadays, mostly unstructured and poorly organized, make this quest ever more pressing, and bring about a range of new challenges. To tackle them, we advocate exploiting a powerful available tool: domain knowledge. We summarize in this paper some research efforts with the broad aim of using *domain knowledge* in order to improve *data access and management*. The domain knowledge we deploy is in the form of *ontologies*, formal descriptions of a domain of interest, expressed as a logical theory in a suitable formalism, like *description logics*.

There are a few efforts aiming to use ontologies in data management, and we focus here on two research directions. First, we consider the problem of answering *ontology mediated queries (OMQs)* where an *ontology* expressing domain knowledge is used to enrich queries to possibly incomplete data. By inferring from the data and the background knowledge implicit facts that contribute to query answers, we can retrieve more complete answers despite the incompleteness of the data. We summarize here some results that we have contributed to, focusing on two directions. The first direction is the design of *expressive OMQ languages*, and in particular, languages that combine the partial open- and closed-world assumptions, allowing to leverage the partial completeness of data when retrieving query answers. We also briefly discuss other expressive languages for writing OMQs that incorporate rich database query formalisms. The second direction we discuss is the development of *algorithms for expressive OMQs*, with focus on *rewriting* techniques.

In the second part of the paper, we make a short review of our work on *reasoning about evolving data*. In that work, we view domain knowledge expressed in ontology languages as *constraints that have to be preserved* in the datasets of interest. We extend a suitable ontology language so that it can describe the effects of *updates* on the data by users or applications, as well as (un)desirable data states that should be reached or avoided when the data changes. This allows us to solve, by reducing them to ontology consistency, relevant *statically analysis tasks* like guaranteeing the preservation of constraints, for *every possible* concrete dataset that may be manipulated by the user or the system.

2 Description Logics and Ontologies

we consider ontologies written in *Description logics (DLs)*, a prominent family of languages for knowledge representation and reasoning. DLs provide the formal underpinning to the *Web Ontology Languages (OWL)* that are the standard for writing and sharing ontologies on the Web [Hitzler *et al.*, 2009], and they are arguably the most popular formalisms for writing ontologies nowadays. Since the OWL languages were standardized over a decade ago, major research efforts have been put into developing DL/OWL ontologies that provide high-quality descriptions of all kinds of domains. To get a general glimpse of these efforts, the reader can visit the community groups hosted by the World Wide Web consortium (W3C) that develop, use, share, and advocate for ontologies in a range of domains as diverse as life sciences and health

care, automotive industry, agriculture, etc.[1] Our research assumes the availability of such ontologies as a stepping stone.

DLs are *decidable fragments* of standard First-order logic (FOL), with a special syntax tailored for convenient knowledge representation. They describe a domain of interest in terms of the relevant *concepts*, which are classes or objects, and properties of those objects, given as *roles* or relations between objects. There is no one DL, but instead, DLs provide a *toolbox* of languages with different but well understood computational properties. They range from the so-called *lightweight* DLs that have limited expressiveness, but allow for efficient inference, to *expressive DLs* that can describe very complex structures, at the cost of high worst-case complexity of reasoning. This toolbox is intended to provide diverse choices for different domains and applications.

Ontologies, called TBoxes in DL jargon, are sets of *axioms* describing the global meaning of terms in our domain. A pair of an ontology $\mathcal{T}$ and a datasets $\mathcal{A}$ (called *ABox* in DLs) is often called a *knowledge base*. As an example, consider the following ontology $\mathcal{T}_i$, in the DL $\mathcal{ALCO}$. It expresses some knowledge about *infection control management and prevention (ICM)*, an important topic in any health care institution.

$$\mathsf{ICM\_spec} \equiv \mathsf{HC\_prof} \sqcap \exists \mathsf{has\_qualif}.\mathsf{ICM\_qual} \quad (1)$$

$$\mathsf{HC\_techn} \sqcup \mathsf{Nurse} \sqcup \mathsf{Doctor} \sqsubseteq \mathsf{HC\_prof} \quad (2)$$

$$\mathsf{Inst\_training} \sqcap \exists \mathsf{has\_topic}.\{ICP\} \sqsubseteq \mathsf{ICM\_qual} \quad (3)$$

Axiom (1) defines ICM specialists as health care professionals who have a qualification in ICM. Axiom (2) says that health care technicians, doctors, and nurses are types of health care professionals, and axiom (3) says that an institutional training on the topic of infection control and prevention is an ICM qualification. We note that an axiom $C \equiv D$ is a shortcut for the pair of axioms $C \sqsubseteq D$ and $D \sqsubseteq C$.

The semantics of DLs is defined in terms of *interpretations*, which are relational structures with a possibly infinite domain, as in FOL. There are only unary and binary relations: concepts are interpreted as sets of objects, and roles as sets of pairs. Informally, given an ontology $\mathcal{T}$ and a dataset $\mathcal{A}$, the *models* of $(\mathcal{T}, \mathcal{A})$ are those structures that extend the facts in $\mathcal{A}$ to satisfy all the axioms in $\mathcal{T}$. For example, consider a dataset $\mathcal{A}_i$ containing the following facts:

$\mathsf{ICM\_spec}(mary)$,
$\mathsf{Nurse}(john)$, $\mathsf{hasQualif}(john, icp\_dipl_1)$,
$\mathsf{Inst\_training}(icp\_dipl_1)$, $\mathsf{hasTopic}(icp\_dipl_1, ICP)$.

In every model $\mathcal{I}$ of $\mathcal{T}_i$ and $\mathcal{A}_i$, both Mary and John are health care professionals and ICM specialists; by the latter, Mary has some ICM qualification, even though we have no details of it. In every interpretation, among others, we have:

$$\{m, j\} \subseteq \mathsf{HC\_prof}^{\mathcal{I}} \quad \{m, j\} \subseteq \mathsf{ICM\_spec}^{\mathcal{I}}$$
$$(m, q_1) \in \mathsf{has\_qual}^{\mathcal{I}} \quad q_1 \in \mathsf{ICM\_qual}^{\mathcal{I}}$$

where m and j respectively denote the objects that interpret $mary$ and $john$ in $\mathcal{I}$, and the 'anonymous' object q_1 stands for Mary's ICM qualification.

[1] https://www.w3.org/community

We note that models are in general not unique, and may be arbitrarily large, or even infinite, not only because the definition of models allows for the existence of arbitrarily many unnecessary objects, but also because many DLs can express knowledge that *enforces* different models or the existence of anonymous objects. For example, if we add to $\mathcal{T}_i$ the converse of axiom (2) saying that all health care professionals are health care technicians, nurses, or doctors:

$$\mathsf{HC\_prof} \sqsubseteq \mathsf{HC\_techn} \sqcup \mathsf{Nurse} \sqcup \mathsf{Doctor}$$

then $(\mathcal{T}_i \mathcal{A}_i)$ has different models where Mary is a doctor, a nurse, or a technician, and they are all relevant for reasoning.

3 Ontology Mediated Query Answering

On of the most successful application areas for ontologies in the last years is the so-called *ontology-based data access (OBDA)* [Lenzerini, 2011], where an ontology acts as common domain conceptualization on top of different, possibly heterogeneous data sources. This has several advantages. By providing a uniform high-level description of the domain, closer to the understanding of the users, it facilitates query formulation. It can enrich the vocabulary, allowing users to query terms defined in the ontology but not explicitely stored in the data. It also allows to use the domain knowledge to infer more informative answers from possibly incomplete data.

An *ontology-mediated query (OMQ)* is usually defined as a pair $(\mathcal{T}, q)$, where $\mathcal{T}$ is an *ontology* (in our case, written in some DL), and q is a query written in a database query language. OMQs are given the *certain answers* semantics, where a tuple is an answer to the query if it is an answer in every model.[2] For example, recall our example above, and consider the following query for finding all ICM specialists.

$$q_1(x) \leftarrow \mathsf{ICM\_spec}(x).$$

The only answer to q_1 over $\mathcal{A}_i$ is $mary$, since there is no other fact $\mathsf{ICM\_spec}(a)$ in $\mathcal{A}_i$. In contrast, the OMQ $(\mathcal{T}_i, q_1)$ allows us to retireve $\{mary, john\}$ from $\mathcal{A}_i$.

Combining complete and incomplete data

DLs and OMQs make the so-called *open-world assumption (OWA)*, which presumes that facts not explicitely given may be either true or false in the models of a given theory. In contrast, the *closed-world assumption (CWA)* assumes that that the data is complete, in the sense that facts are false unless explicitely made true by the theory. The OWA is the fundamental semantic feature that makes DL ontologies and OMQs adequate for describing incomplete data. However, there is a downside: *all* data is viewed as incomplete, and this may also result in fewer answers than expected. For example, suppose that a hospital assigns ICM specialists to regularly audit the infection prevention practices of the different divisions. Let $\mathcal{A}_i'$ be $\mathcal{A}_i$ extended with the following assignment data:

$\mathsf{Dept}(dept_1)$, $\mathsf{Dept}(dept_2)$,
$\mathsf{has\_suborg}(dept_1, div_1)$, $\mathsf{has\_suborg}(dept_2, div_3)$,
$\mathsf{has\_suborg}(dept_1, div_2)$,
$\mathsf{audits}(mary_1, div_1)$, $\mathsf{audits}(john_2, div_3)$,
$\mathsf{audits}(john_1, div_2)$.

[2] On one given model, the semantics of the queries is as in a regular relational database.

Let $\mathcal{T}_i'$ be the ontology $\mathcal{T}_i$ extended with the following axioms, stating that a department is fully audited if all its divisions have an auditor assigned:

$$\exists \mathsf{audits}.\mathsf{ICM\_spec} \sqsubseteq \mathsf{ICM\_audited} \quad (4)$$

$$\mathsf{Dept} \sqcap \forall \mathsf{has\_suborg}.\mathsf{ICM\_audited} \sqsubseteq \mathsf{audited\_dept} \quad (5)$$

The following query asks for fully audited departments:

$$q_2(x) \leftarrow \mathsf{audited\_dept}(x)$$

If we know that all the departments and their divisions are explicitely listed in the database, we expect both $dept_1$ and $dept_2$ as query answers. Unfortunately, the OMQ $(\mathcal{T}_i', q_2)$ does not have any answers over the dataset $\mathcal{A}_i'$, since the existence of additional divisions of $dept_1$ and $dept_2$ cannot be excluded, and there are models of $(\mathcal{T}_i', \mathcal{A}_i')$ where other, non-audited departments exist. In most application domains, complete and incomplete information coexist. Knowing that the data is complete in some way is also a form of domain knowledge that could be used to infer more informative answers, but this is not possible with standard OMQs. In our recent work, we have explored OMQ languages that can express that certain relations are complete, by enriching a standard OMQ $(\mathcal{T}, q)$ with a set Σ of *closed predicates*. The latter are interpreted under the CWA: there can be no more facts in their extension than those explicitely stated in the current dataset $\mathcal{A}$. In our example, if has_suborg is considered closed, the models of $(\mathcal{T}_i', \mathcal{A}_i')$ are only those where has_suborg contains exactly the pairs listed above, and both $dept_1$ and $dept_2$ become query answers, as desired.

OMQs with closed predicates where first considered in [Seylan *et al.*, 2009; Franconi *et al.*, 2011], and interest in them has resurged in the last few years. Some authors have focused on their *data complexity*, which is the complexity of OMQ evaluation measured in terms of size of the dataset $\mathcal{A}$ only, assuming that $\mathcal{T}$ and q are fixed [Lutz *et al.*, 2013; Lutz *et al.*, 2015]. In our research, we have studied the the overall effect of closed predicates in the *combined complexity* of reasoning, obtaining tight bounds for both lightweight and expressive DLs [Ngo *et al.*, 2016]. As discussed below, we have also developed *rewriting techniques* for OMQs with closed predicates, and expressive OMQ languages that combine ontologies and *rules* with open and closed predicates.

Expressive OMQ languages

The most popular language for writing OMQs puts together ontologies of the so-called DL-Lite family[Calvanese *et al.*, 2007], and *conjunctive queries* or unions thereof (UCQs). This is an appealing combination: UCQs are among the basic database query languages, and at the heart of formalisms like SQL, while DL-Lite was explicitely tailored for describing data sources and allowing efficient inference. However, the combination of DL-Lite and UCQs has many stark limitations. Indeed, many domains call for expressive features present in standard ontology languages, but not in DL-Lite. This could include, for example, stating that relations like part_of, or has_suborg in our example, are transitive. Also universal restrictions, as in axiom (5), are often useful but they are not supported by DL-Lite and other lightweight DLs. On the query side, UCQs also have limited expressiveness. In particular, they lack *recursion* as needed, for example, in *reachability queries* that can flexibly navigate paths of different length in our data to find the desired information. For example, the following query q_3 is a so-called *conjunctive regular path query (CRPQ)* that finds all sub-organizations of a hospital h, no matter their level, that are audited by a nurse:

$$q_3(x) \leftarrow \exists y.\ \mathsf{has\_suborg}^*(h, x) \wedge \mathsf{audits}(y, x) \wedge \mathsf{Nurse}(y).$$

Much of our work has focused on expressive OMQs comprising more expressive DLs than DL-Lite, more expressive query languages than UCQs, or additional features like closed predicates. Among our specific focus areas are queries with recursive features like CRPQs and their variations e.g.,[Calvanese *et al.*, 2009; Calvanese *et al.*, 2011; Calvanese *et al.*, 2014; Bienvenu *et al.*, 2015; Bienvenu *et al.*, 2014; Ortiz *et al.*, 2011]. We have tried to understand the limits and possibilities of the different combinations, established complexity results, and developed algorithms for them. For more detailed discussion, the reader can refer to the surveys in [Ortiz and Simkus, 2012; Bienvenu and Ortiz, 2015] and their references. Here we briefly discuss only one recent proposal.

The so-called *hybrid languages* that combine DL ontologies with *rule formalisms* can be viewed as OMQs with a more expressive query component. Here predicates are usually partitioned into ontology predicates, interpreted under the OWA, and predicates whose extension is determined by the rules. Syntactic restrictions (e.g., on variable occurrences) are needed to preserve decidability [Rosati, 2007]. We recently introduced a rich hybrid language called *Clopen Knowledge Bases (CKBs)* that combines DL ontologies with Answer Set Programming (ASP), a rich formalism that extends recursive Datalog rules with negation as failure. CKBs generalize prominent hybrid languages like *r-hybrid* [Rosati, 2006b] and $\mathcal{DL}$*+Log* [Rosati, 2006a]. Their most novel feature is the freedom to partition predicates into open and closed, making them better suited for domains where complete and incomplete information coexist. CKBs shed light on the relationship between hybrid KBs and closed predicates, showing that the distinction between CWA and OWA predicates does not need to be tied to whether a predicate occurs in the ontology or not. We showed that a large class of CKBs with $\mathcal{ALCH}$ ontologies can be translated into a plain ASP program, and thus inference can be realized using existing ASP solvers.

Rewriting techniques

In the context of OMQs, *query rewriting* is the following problem. Assume that a standard database query language $\mathcal{L}$ has been fixed as *target language*. Given an OMQ $Q = (\mathcal{T}, q)$, obtain a query q' in $\mathcal{L}$ such that, for any dataset $\mathcal{D}$, the certain answers to Q and q' coincide. Such rewritings are central in OMQ research. On the one hand, their existence and size sheds light on the expressiveness and succinctness of OMQ languages. On the other hand, rewritings may make it possible to implement OMQA systems by reusing efficient existing technologies for the target language. Many rewritings for OMQs are known; for an overview, see the references

in [Bienvenu, 2016; Bienvenu and Ortiz, 2015]. We briefly recall one of recent contributions in this area.

We consider OMQs with closed predicates, where the ontology is in the expressive DL $\mathcal{ALCHIO}$. Answering such OMQs is intractable in data complexity, and ExpTime hard in combined complexity. Moreover, closed predicates make the query language *non-monotonic*. Recall the OMQ with closed predicates $Q = (\mathcal{T}_i', \{\mathsf{has\_suborg}\}, q_2)$ discussed above. While $dept_2$ is in the answers to Q over $\mathcal{A}_i'$, it is not an answer over $\mathcal{A}_i'' = \mathcal{A}_i' \cup \{\mathsf{has\_suborg}(dept_2, div_4)\}$. This indicates that the target language needs to support non-monotonic queries. We proposed a rewriting into *disjunctive Datalog with (stratified) negation as failure* [Ahmetaj *et al.*, 2016]. The most relevant feature of this rewriting is its size: it is polynomially bounded in the input, while most similar rewriting approaches, even for less expressive OMQs, need exponential time and generate an exponentially larger query.

4 Managing Evolving Data

In a different line of research, we have advocated the use of DL ontologies to reason about the *evolution* of data that is modified by users or applications [Ahmetaj *et al.*, 2017]. We have focused on the paradigm of graph-structured data (GSD) [Sakr and Pardede, 2011], increasingly advocated as a more flexible alternative to traditional relational databases, useful in settings where there is no rigid schema fixed a priori, like Web data. In GSD, information is stored as node- and edge-labeled graphs, where labels carry semantic information; such graphs are no different from the relational structures that we have defined as *interpretations* of DLs. This makes DLs obvious candidates for describing properties of GSD. As our work shows, we can leverage DLs to obtain algorithms for challenging data management problems, such as testing whether the satisfaction of integrity constraints is preserved after a sequence of updates *for every possible data instance*. This goes way beyond checking whether for a given initial data instance the constraints are preserved, which is the verification problem usually considered for dynamic data-centric systems.

In our work, we view GSD as finite DL interpretations. With this view, standard DLs can naturally model rich integrity constraints on the data. They are also naturally suitable to specify desirable and undesirable conditions of data states, which should be ensured or avoided during data evolution. However, we also use DLs for for describing the actions that can be used to modify the data, and the effect of those actions. This use is less standard, but can also be done in suitable DLs that support some less frequent constructs. We consider both a very expressive DL, and a lightweight DL. Those DLs can, for example, describe the action of reassigning an ICM auditor from one unit to another. We establish decidability (and in most cases, also tight complexity bounds) for a range of decision problems related to the evolution of GSD, by reducing the problems to testing satisfiability of a (standard, static) knowledge base. These problems include the following:

Static verification Given a set of constraints and a sequence of actions, is it the case that, for every possible data state initially satisfying the constraints, the constraints are still satisfied after the execution of the actions?

Plan verification Does a given sequence of actions necessarily/possibly lead to a state where some property (either desired or undesired) holds? We study these problem for a given initial dataset, and also for the case in only a partial description of the initial state is available.

Plan synthesis Is there a sequence of actions chosen from a given set (i.e., a *plan*), that leads the data to a into a state where some property (either desired or undesired) holds? Again, we study some variations: for a given initial dataset, or for a partial description of it, as well as for plans of bounded length.

In our example, one could have constraints that say that a specialist cannot be assigned as auditor to more than three units, or across different departments. One could then verify whether a given re-assignment transaction is designed in such a way that is always preserves these constraints when applied to any dataset that initially satisfies them.

While DLs allow to flexibly and naturally express constraints on GSD, the DLs considered in [Ahmetaj *et al.*, 2017] have limited means to express navigation on graphs, and in particular, they do not capture the some well-known *path constraint* languages for GSD. For this reason, in follow up work we extended the techniques and results to related DLs, but with the so-called *regular role expressions* [Calvanese *et al.*, 2016]. Although some results were negative (e.g., the use of paths in actions easily leads to undecidability), many interesting cases remain decidable and their computational cost is as low as one could hope for. As a side result, by capturing path constraints in DLs, we obtained an alternative proofs of decidability, sometimes with significantly improved complexity bounds, for path constraint reasoning problems previously studied in the literature. This provides further evidence of the suitability of DLs as flexible tools for data management.

5 Closing Remarks

We firmly believe that the domain knowledge captured in ontologies has a great potential for improving the quality and reliability of data management and information systems. There have been some success stories, most notably that of OBDA, which in roughly a decade has gone from the basic research to industrial-level readiness as a data integration paradigm. However, there is much more that can be done towards enabling smart, reliable, data-centric systems using domain knowledge. Breaking the long-standing wall between the open- and closed-world view, between complete and incomplete data, is in my opinion, one of the most exciting challenges in the short term. We are also enthusiastic about using ontologies to reason about evolving data. The possibility of carrying out static verification of transactions effectively, using DL reasoners, is promising, and can help pave the way to more trustable, provably verified data management systems. Hopefully, this is one of the many ways in which ontologies can be used to make our information systems better.

Finally, I would like to close this paper with an invitation to work together, as AI researchers, towards information systems that are trustable, fair, ethical, and that make our lives better. As a researcher in knowledge representation, I believe that domain knowledge has the potential to help us get closer

to that goal. However, domain knowledge alone will clearly not suffice, we need join our efforts to make sure that emerging information technologies remain a force for good and contribute to a better society.

Acknowledgments

The presented work was done jointly with several co-authors, to whom I am very grateful. It has been supported by the Austrian Science Fund (FWF) via the now finished project T515, and the ongoing P30360, P30873, and W1255.

References

[Ahmetaj *et al.*, 2016] Shqiponja Ahmetaj, Magdalena Ortiz, and Mantas Simkus. Polynomial datalog rewritings for expressive description logics with closed predicates. In *IJCAI*, pages 878–885. AAAI Press, 2016.

[Ahmetaj *et al.*, 2017] Shqiponja Ahmetaj, Diego Calvanese, Magdalena Ortiz, and Mantas Simkus. Managing change in graph-structured data using description logics. *ACM Trans. Comput. Log.*, 18(4):27:1–27:35, 2017.

[Bienvenu and Ortiz, 2015] Meghyn Bienvenu and Magdalena Ortiz. Ontology-mediated query answering with data-tractable description logics. In *Reasoning Web*, volume 9203 of *Lecture Notes in Computer Science*, pages 218–307. Springer, 2015.

[Bienvenu *et al.*, 2014] Meghyn Bienvenu, Diego Calvanese, Magdalena Ortiz, and Mantas Simkus. Nested regular path queries in description logics. In *KR*. AAAI Press, 2014.

[Bienvenu *et al.*, 2015] Meghyn Bienvenu, Magdalena Ortiz, and Mantas Simkus. Regular path queries in lightweight description logics: Complexity and algorithms. *J. Artif. Intell. Res.*, 53:315–374, 2015.

[Bienvenu, 2016] Meghyn Bienvenu. Ontology-mediated query answering: Harnessing knowledge to get more from data. In *IJCAI*, pages 4058–4061. IJCAI/AAAI Press, 2016.

[Calvanese *et al.*, 2007] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Automated Reasoning*, 39(3):385–429, 2007.

[Calvanese *et al.*, 2009] Diego Calvanese, Thomas Eiter, and Magdalena Ortiz. Regular path queries in expressive description logics with nominals. In *IJCAI*, pages 714–720, 2009.

[Calvanese *et al.*, 2011] Diego Calvanese, Magdalena Ortiz, and Mantas Simkus. Containment of regular path queries under description logic constraints. In *IJCAI*, pages 805–812. IJCAI/AAAI, 2011.

[Calvanese *et al.*, 2014] Diego Calvanese, Thomas Eiter, and Magdalena Ortiz. Answering regular path queries in expressive description logics via alternating tree-automata. *Inf. Comput.*, 237:12–55, 2014.

[Calvanese *et al.*, 2016] Diego Calvanese, Magdalena Ortiz, and Mantas Simkus. Verification of evolving graph-structured data under expressive path constraints. In *ICDT*, volume 48 of *LIPIcs*, pages 15:1–15:19. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[Franconi *et al.*, 2011] Enrico Franconi, Yazmin Angélica Ibáñez-García, and Inanç Seylan. Query answering with dboxes is hard. *Electr. Notes Theor. Comput. Sci.*, 278:71–84, 2011.

[Hitzler *et al.*, 2009] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph, editors. *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 2009. 27 October 2009, Available at http://www.w3.org/TR/owl2-primer/.

[Lenzerini, 2011] Maurizio Lenzerini. Ontology-based data management. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 5–6, New York, NY, USA, 2011. ACM.

[Lutz *et al.*, 2013] Carsten Lutz, Inanç Seylan, and Frank Wolter. Ontology-based data access with closed predicates is inherently intractable(sometimes). In *IJCAI*, pages 1024–1030. IJCAI/AAAI, 2013.

[Lutz *et al.*, 2015] Carsten Lutz, Inanç Seylan, and Frank Wolter. Ontology-mediated queries with closed predicates. In *IJCAI*, pages 3120–3126. AAAI Press, 2015.

[Ngo *et al.*, 2016] Nhung Ngo, Magdalena Ortiz, and Mantas Simkus. Closed predicates in description logics: Results on combined complexity. In *AMW*, volume 1644 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.

[Ortiz and Simkus, 2012] Magdalena Ortiz and Mantas Simkus. Reasoning and query answering in description logics. In *Reasoning Web*, volume 7487 of *Lecture Notes in Computer Science*, pages 1–53. Springer, 2012.

[Ortiz *et al.*, 2011] Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Query answering in the horn fragments of the description logics SHOIQ and SROIQ. In *IJCAI*, pages 1039–1044. IJCAI/AAAI, 2011.

[Rosati, 2006a] Riccardo Rosati. Dl+log: Tight integration of description logics and disjunctive datalog. In *KR*, pages 68–78. AAAI Press, 2006.

[Rosati, 2006b] Riccardo Rosati. The limits and possibilities of combining description logics and datalog. In *RuleML*, pages 3–4. IEEE Computer Society, 2006.

[Rosati, 2007] Riccardo Rosati. The limits of querying ontologies. In *ICDT*, volume 4353 of *Lecture Notes in Computer Science*, pages 164–178. Springer, 2007.

[Sakr and Pardede, 2011] Sherif Sakr and Eric Pardede, editors. *Graph Data Management: Techniques and Applications*. IGI Global, 2011.

[Seylan *et al.*, 2009] Inanç Seylan, Enrico Franconi, and Jos de Bruijn. Effective query rewriting with ontologies over dboxes. In *IJCAI*, pages 923–925, 2009.