# Rewritability Results for OMQs with Closed Predicates [⋆]

Magdalena Ortiz

Institute of Logic and Computation, TU Wien, Austria
ortiz@kr.tuwien.ac.at

**Abstract.** The semantics of Description Logic (DL) ontologies adopts the *open-world assumption (OWA)* from classical first-order logic (FOL), which makes them well suited to reason about incomplete knowledge. However, there are many applications where incomplete and complete data co-exist, and the standard OWA, which assumes that all data is incomplete, allows to draw less inferences than desired. Adding *closed-predicates* to ontology-mediated queries (OMQs) is an elegant and powerful way to reach better conclusions leveraging partial completeness, but unfortunately, it is well-known that OMQs with closed predicates are coNP-hard in practically every non-trivial setting, and hence they are not FOL-rewritable. We recall here a small selection of recent results on rewritability of OMQs with closed predicates: two polynomial rewritings into extensions of Datalog for restricted classes of conjunctive queries mediated by ontologies in very expressive DLs like $\mathcal{ALCHOI}$ and $\mathcal{ALCHOIF}$, and a recent approach that identifies an FOL-rewritable family of OMQs that includes full conjunctive queries and DL-Lite ontologies, while restricting in a novel yet non-trivial way the use of closed predicates.

## 1 Motivation and Background

Like classical first-order logic (FOL), *description logics* (DLs) adopt the *open-world assumption (OWA)* [5]. A model of a DL ontology $\mathcal{O}$ is a structure that satisfies the axioms in $\mathcal{O}$, and which can freely interpret as true or false the assertions whose truth or falsity is not implied by $\mathcal{O}$. A DL ontology typically has many models, many of which can contain facts that are not relevant to the modeled problem but are not ruled out by the ontology. With this classical semantics, DLs are well suited to reason about incomplete knowledge, and they have been successfully applied in many areas where problems of interest can be modeled and solved by means of different reasoning services, which are usually defined in terms of logical entailment in a way that 'irrelevant models' are immaterial.

An important reasoning service that is found at the core of many modern applications of DLs (such as the prominent *ontology-based data access (OBDA)*

[29] paradigm) is that of *ontology-mediated query (OMQ) answering* [8], that allows is to retrieve all certain answers to a given query over a possibly incomplete dataset—or *ABox*, in DL jargon—taking into account not only the explicit data, but also all additional facts that can be inferred from the data using the ontology. In an OMQ we pair a given query $q$ with a domain ontology $\mathcal{O}$. To evaluate it over a possibly incomplete dataset $\mathcal{A}$, we adopt the *certain answer semantics*, where a tuple is an answer to $(q, \mathcal{O})$ if it is an answer to $q$ in *every model* of $\mathcal{A}$ and $\mathcal{O}$.

*Example 1.* Consider the following example, inspired by Example 1 in [25]. Suppose that we have a possibly incomplete dataset $\mathcal{A}$ of car models and their motors (e.g., in an online car marketplace).

$$\begin{array}{llll}
\mathsf{Car}(c1). & \mathsf{hasMotor}(c1, \text{2KD}). & \mathsf{DieselMotor}(\text{2KD}). & \mathsf{ToyotaMotor}(\text{2KD}).\\
\mathsf{Car}(c2). & \mathsf{hasMotor}(c3, \text{2SDI}). & \mathsf{DieselMotor}(\text{2SDI}). & \mathsf{ToyotaMotor}(\text{1GZ}).\\
\mathsf{ToyotaCar}(c2). & & \mathsf{PetrolMotor}(\text{1GZ}). & \\
\mathsf{Car}(c3). & & &
\end{array}$$

To find all cars with an internal combustion motor, one can leverage an ontology $\mathcal{O}$ that includes, among others, the following axioms stating that petrol and diesel motors are internal combustion (IC) motors, and Toyota cars have Toyota motors.

$$\begin{array}{ll}
\mathsf{PetrolMotor} \sqsubseteq \mathsf{ICMotor} & \mathsf{ToyotaCar} \sqsubseteq \exists \mathsf{hasMotor}.\mathsf{ToyotaMotor}\\
\mathsf{DieselMotor} \sqsubseteq \mathsf{ICMotor} &
\end{array}$$

Let $q(x) = \exists y.\ \mathsf{Car}(x) \wedge \mathsf{hasMotor}(x, y) \wedge \mathsf{ICMotor}(y)$. Then, by posing the OMQ $(q, \mathcal{O})$ over $\mathcal{A}$, one obtains as certain answers $c1$ and $c3$, the two cars that are known to have an IC motor. Note that we know that $c2$ has a Toyota motor, but we do not know whether it is an IC motor.

In the OMQ community, *rewritability* results play a fundamental role (see, e.g., [2, 4, 9, 11, 20] and their references). We say that a family of OMQs $\mathcal{Q}$ is *rewritable into a target query language* $\mathcal{L}$, if for every $Q = (q, \mathcal{O})$ in $\mathcal{Q}$, there is a query $\mathit{rew}(Q)$ in $\mathcal{L}$ such that, for every $\mathcal{A}$, the certain answers of $\mathcal{Q}$ over $\mathcal{A}$ coincide with the (usual) answers of $\mathit{rew}(Q)$ over $\mathcal{A}$. The rewritability into a target $\mathcal{L}$ means—at least in principle—that OMQ answering in the class $\mathcal{Q}$ can be realized using off-the-shelf query answering engines for $\mathcal{L}$, and indeed, rewritings have played a crucial role the successful implementation of OMQs. FOL-rewritability is particularly appealing, since it means that OMQ answering can be realized using existing RDBMSs. For example, for the OMQ above we can use the following FOL-rewriting that retrieves the same answers when evaluated over $\mathcal{A}$, without the need to reason about $\mathcal{O}$.

$$\begin{aligned}
\mathit{rew}(q, \mathcal{O})(x) = \exists y\ & \mathsf{Car}(x) \wedge \mathsf{hasMotor}(x, y) \wedge\\
& \big(\mathsf{ICMotor}(y) \vee \mathsf{DieselMotor}(y) \vee \mathsf{PetrolMotor}(y)\big)
\end{aligned}$$

The importance of rewritings is not restricted to reusing technologies for OMQ answering. In fact, many existing rewriting algorithms produce rewritings that are too large or complex to evaluate with standard query engines. However, even if this is the case, rewritings can play an important role also in foundational research. Establishing the (non-)existence of rewritings, as well as bounds on their sizes, are central tools to study the relative expressiveness of different OMQ languages, and to compare them to traditional query languages.

## 1.1 OMQs with Closed Predicates

There are many applications of DLs in general, and of OMQs in particular, where the OWA results in a semantics that is weaker than desired, as intended consequences are not entailed because they are falsified in some unintended model [2, 26]. For example, if *all Toyota motors are IC motors*, then we want to obtain in the example above also $c2$ as an answer. To overcome this challenge, variations of standard DLs with different forms of (partial) *closed-world assumption* (CWA) have been extensively studied since the early days of DLs and keep receiving significant attention, e.g., [6, 10, 12, 14–18, 21, 27]. In the setting of OMQ answering, we may find ourselves facing a scenario where the data is incomplete, but some parts of it are known to be complete, and standard OMQs do not have the expressive means to harness the partial data completeness for obtaining better inferences.

*Closed predicates* are a simple and appealing way to evaluate OMQs in settings where incomplete and complete information co-exist. An *OMQ with closed predicates (OMQCs)* $(q, \mathcal{O}, \Sigma)$ extends an an $(q, \mathcal{O})$ with a set of predicates $\Sigma$ that are to be viewed as complete in the data [15, 2, 26]. The certain answers $(q, \mathcal{O}, \Sigma)$ over $\mathcal{A}$ are thus the tuples that are an answer to $q$ in every *intended* model of $\mathcal{A}$ and $\mathcal{O}$, that is, every structure extending $\mathcal{A}$ to satisfy $\mathcal{O}$, but without extending the extensions of the predicates in $\Sigma$.

*Example 2.* Assume that the list of Toyota motors in $\mathcal{A}$ is imported from a complete list provided by the manufacturer.[1] We can express this information by saying that $\mathsf{ToyotaMotor}$ is a closed predicate, and harness this information to retrieve also $c2$. The set of certain answers to the OMQ $(q, \mathcal{O}, \{\mathsf{ToyotaMotor}\})$ over $\mathcal{A}$ is $\{c1, c2, c3\}$.

Several authors have studied OMQCs [2, 15, 24–26, 28], but unfortunately, most works are marked by negative complexity results. Indeed, reasoning with closed predicates has a high computational cost [15, 24, 28]. In the setting of OMQs, the biggest obstacle is that closed predicates cause intractability in data complexity: OMQC answering is coNP-hard, even when the ontology is in very weak DLs like DL-Lite$_{\text{core}}$ or $\mathcal{EL}$, and the query is in very restricted classes of *conjunctive queries (CQs)* [15, 26]. This, of course, immediately implies that there do not exist any *data independent rewritings of OMQCs* into query languages with tractable data complexity, such as FOL or Datalog. Only severely restricted classes of CQs and DL-Lite ontologies have been shown to be rewritable

[1] Our toy example has only two motors, but a complete database of all Toyota motors can be easily found online.

into equivalent FOL queries [25, 26]. Moreover, for DL-Lite it has been shown that a class of OMQCs is FOL-rewritable only if it satisfies a *convexity condition* which, as it turns out, implies that the presence of closed predicates is irrelevant [24]. For a fine-grained non-uniform analysis of what makes OMQC answering (in)tractable, please see [26].

However, despite this rather discouraging landscape, we have seen interesting progress in rewritability results for OMQCs in the last couple of years, e.g., [2, 3, 7, 19, 23, 25]. Here we will review a selection of these works, focusing on two types of results. First, we discuss two polynomial rewritings for OMQCs that allow for very expressive DLs as ontology languages, and use as target languages variants of Datalog. Then we will briefly recall a recent approach that identifies an FOL-rewritable family of OMQs that includes full CQs and DL-Lite ontologies, but restricts in a novel yet non-trivial way the use of closed predicates.

## 2 Rewriting instance OMQCs in very expressive DLs

Since we know that OMQC answering is coNP-hard in basically all interesting cases [24], it makes sense to consider as target for rewritings more expressive query languages that also exhibit such data complexity.

Moreover, *OMQCs are a non-monotonic query language.*

*Example 3.* Recall that the certain answers to $(q, \mathcal{O}, \{\mathsf{ToyotaMotor}\})$ over $\mathcal{A}$ are $\{c1, c2, c3\}$. Assume that we received and updated list of Toyota motors that includes $\mathsf{ToyotaMotor}(1\mathrm{PMSM})$, which is not listed as an IC motor. The certain answers to $(q, \mathcal{O}, \{\mathsf{ToyotaMotor}\})$ over $\mathcal{A} \cup \{\mathsf{ToyotaMotor}(1\mathrm{PMSM})\}$ are now $\{c1, c3\}$, since $c2$ can now have 1PMSM as motor and thus no IC motor.

Therefore, any potential rewriting must consider as target a query language that is also non-monotonic. This makes *extensions of Datalog with negation* a natural target [13].

Indeed, rewritings into Datalog with negation have been developed for very expressive DLs like $\mathcal{ALCHOI}$ [2] and $\mathcal{ALCHOIF}$ [19], considering as query languages *instance queries (IQs)* and other restricted classes of CQs. Crucially, both works produce Datalog programs of *polynomial size*. We also stress that both of these translations are *data independent*, hence the presence of nominals in the underlying DL does not help us simulate the closed predicates.

In the former work [2], we characterize the satisfaction of the ontology as a two-player game (inspired by *type elimination* algorithms), and then we design a Datalog query that can decide the existence of a winning strategy for the game. The resulting Datalog program uses a restricted form of negation and falls in a fragment that can be evaluated in deterministic single exponential time, hence it also provides a query answering algorithm that is worst-case optimal in *combined complexity*. If there are no closed predicates—that is, for an instance query mediated by a plain $\mathcal{ALCHOI}$ ontology—the translation yields a positive disjunctive Datalog program of polynomial size, and if the input is in a *Horn* fragment, a modified technique produces a plain Datalog program. These ideas have been used also for *tuple-generating dependencies (TGDs)* in the absence of closed predicates, showing that guarded TGDs can be polynomially encoded into Datalog, and disjunctive guarded TGDs into disjunctive Datalog [1].

For the DL $\mathcal{ALCHOIF}$, in contrast, IQ answering (without closed predicates) is already co-NEXPTIME-hard, and the ideas discussed above are not applicable. Therefore in our most recent work [19] we characterize models in terms of *matching set of mosaics*, and show that the existence of the latter can be reduced to integer programming. Then we define a carefully crafted program in Datalog with negation that finds solutions to dynamically generated systems of integer inequalities. This translation yields optimal co-NP and co-NEXPTIME-hard upper bounds in data and combined complexity, respectively. We note that the latter bound is new, and it was not obvious whether such a bound was possible in the presence of closed predicates, and of the tricky combination of nominals, inverse roles and role functionality.

Both rewritings consider IQs and restricted families of CQs. Unfortunately, allowing full CQs precludes the existence of a polynomial sized rewritings under standard complexity assumptions, even if we restrict the ontology language. Indeed, in the presence of closed predicates, OMQ answering with CQs is 2EXPTIME-hard already for $\mathcal{EL}$ and DL-Lite$_\mathcal{R}$ [28], while entailment in Datalog with negation is in coNEXPTIME$^{\text{NP}}$.

## 3 An FOL-rewritable family of ontology-mediated CQs

Finally, we discuss a recent positive FOL-rewritability result that applies to full CQs. We have already made clear that OMQCs are not FOL-rewritable for any standard DL. However, we identified an interesting setting where OMQCs that pair arbitrary CQs and DL-Lite$_\mathcal{R}$ ontologies are FOL-rewritable, but the use of closed predicates is restricted: these may not occur in the ontology, but instead, we allow *complex ABox assertions* where closed predicates may occur [3]. Although somewhat non-standard at first sight, such a setting is simple and intuitive, and captures the typical examples that are used to argue in favor of closed predicates in OMQs.

The key idea underlying this rewriting is is that, if the closed predicates do not occur in the TBox, then one may need to do some form of case-by-case reasoning over models, but this can be encoded into a (non-monotonic) FOL query that uses universal quantification to iterate over all possible extensions of the closed predicates in the models of each input dataset. The resulting query can be large and take a rather complex form, but does not seem to fully rule out the possibility of implementation.

*Example 4.* Consider a modified version of the example above, which is in fact a minor modification of Example 1 in [25]. Since $\mathsf{ToyotaMotor}$ is closed, it is disallowed in the ontology, and hence we consider $\mathcal{O}'$:

$$\mathsf{PetrolMotor} \sqsubseteq \mathsf{ICMotor} \qquad \mathsf{DieselMotor} \sqsubseteq \mathsf{ICMotor}$$

We can, however, explicitely say that $c2$ has a Toyota motor by assuming the ABox assertion $\exists\mathsf{hasMotor}.\mathsf{ToyotaMotor}(c2)$. The techniques in [3] essentially rewrites the OMQ $(q, \mathcal{O})$ from Example 1 into (an equivalent form of) the FOL query:

$$\begin{aligned}
&\exists y\ \mathsf{Car}(x) \wedge \mathsf{hasMotor}(x, y) \wedge \\
&\quad \big(\mathsf{ICMotor}(y) \vee \mathsf{DieselMotor}(y) \vee \mathsf{PetrolMotor}(y) \vee \\
&\quad (\mathsf{ToyotaMotor}(y) \wedge \forall y' (\mathsf{ToyotaMotor}(y')) \to \\
&\quad (\mathsf{ICMotor}(y') \vee \mathsf{DieselMotor}(y') \vee \mathsf{PetrolMotor}(y'))))\big)
\end{aligned}$$

Essentially, the disjunction in $\mathit{rew}(q, \mathcal{O})(x)$ now has an additional disjunct that allows to substitute $y$ by any Toyota motor in every dataset in which all Toyota motors are IC motors. Note that this query correctly retrieves $\{c1, c2, c3\}$ when evaluated over $\mathcal{A}$, but only $\{c1, c3\}$ when evaluated over

$$\mathcal{A} \cup \{\mathsf{ToyotaMotor}(\mathrm{1PMSM})\}.$$

The similar version of this example given in [25] is not FOL-rewritable: the axiom $\mathsf{ToyotaCar} \sqsubseteq \exists\mathsf{hasMotor}.\mathsf{ToyotaMotor}$ in the ontology makes it *unsafe*.

We note that the algorithm in [3] considers a slightly more general framework. Rather than adding complex ABox assertions, one can make a set of *hypothesis* (which are instances of certain *assumption patterns* that are complex concepts given as part of the input). The output for such a query are *conditional answers* that pair instantiated hypothesis with tuples that are answers given the hypothesis.

## 4 Outlook

Reconciling the open- and closed-world assumptions is one of the most fascinating challenges that the OMQ and DL communities face. Much has been achieved, but we need to pay more attention to understanding the way complete and incomplete information coexist in real-world applications. There is plenty of room for meaningful characterizations of *sets of intended models* that are more useful that the full set of open-world models that is prevalent in DL reasoning, without fully giving up the open-world inference and the possibility of reasoning about incomplete aspects of the world [18]. Many great ideas have been proposed over the decades in areas like *hybrid languages* that combine rules and ontologies (e.g., [7, 27, 22]), and in *circumscription* [10] and other forms of non-monotonic reasoning in DLs [14, 6]. However, some of these ideas have not been pursued sufficiently or left aside for too long.

**Acknowledgments** The works reported here were supported by the Austrian Science Fund (FWF) projects P30360, P30873 and W1255. They were carried out with different co-authors, to whom I am grateful: special thank you to Shqiponja Ahmetaj, Medina Andreşel, Sanja Lukumbuzya, and Mantas Šimkus.

## References

1. Ahmetaj, S., Ortiz, M., Simkus, M.: Rewriting guarded existential rules into small datalog programs. In: Kimelfeld, B., Amsterdamer, Y. (eds.) 21st Int. Conf. on Database Theory, ICDT 2018. LIPIcs, vol. 98, pp. 4:1–4:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018). https://doi.org/10.4230/LIPIcs.ICDT.2018.4, https://doi.org/10.4230/LIPIcs.ICDT.2018.4
2. Ahmetaj, S., Ortiz, M., Simkus, M.: Polynomial rewritings from expressive description logics with closed predicates to variants of datalog. Artificial Intelligence **280**, 103220 (2020). https://doi.org/10.1016/j.artint.2019.103220, https://doi.org/10.1016/j.artint.2019.103220
3. Andresel, M., Ortiz, M., Simkus, M.: Query rewriting for ontology-mediated conditional answers. In: Proc. of the 34th AAAI Conf. on Artificial Intelligence (AAAI 2020). pp. 2734–2741. AAAI Press (2020), https://aaai.org/ojs/index.php/AAAI/article/view/5660
4. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The dl-lite family and relations. J. Artif. Int. Res. **36**(1), 1–69 (Sep 2009), http://dl.acm.org/citation.cfm?id=1734953.1734954
5. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
6. Baader, F., Hollunder, B.: Embedding defaults into terminological knowledge representation formalisms. J. of Automated Reasoning **14**(1), 149–180 (1995). https://doi.org/10.1007/BF00883932, https://doi.org/10.1007/BF00883932
7. Bajraktari, L., Ortiz, M., Simkus, M.: Combining rules and ontologies into clopen knowledge bases. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proc. of the 32nd AAAI Conf. on Artificial Intelligence (AAAI 2018). pp. 1728–1735. AAAI Press (2018), https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/ 16991
8. Bienvenu, M.: Ontology-mediated query answering: Harnessing knowledge to get more from data. In: Kambhampati, S. (ed.) Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI 2016). pp. 4058–4061. IJCAI/AAAI Press (2016), http://www.ijcai.org/Abstract/16/600
9. Bienvenu, M., Kikot, S., Kontchakov, R., Podolskii, V.V., Zakharyaschev, M.: Ontology-mediated queries: Combined complexity and succinctness of rewritings via circuit complexity. J. ACM **65**(5), 28:1–28:51 (2018). https://doi.org/10.1145/3191832, https://doi.org/10.1145/3191832
10. Bonatti, P.A., Lutz, C., Wolter, F.: The complexity of circumscription in dls. J. Artif. Intell. Res. **35**, 717–773 (2009)
11. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. J. Autom. Reasoning **39**(3), 385–429 (2007)
12. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Eql-lite: Effective first-order query processing in description logics. In: Veloso, M.M. (ed.) Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI 2007) (2007), http://ijcai.org/Proceedings/07/Papers/042.pdf
13. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Comput. Surv. **33**(3), 374âĂŞ425 (Sep 2001). https://doi.org/10.1145/502807.502810, https://doi.org/10.1145/502807.502810

14. Donini, F.M., Nardi, D., Rosati, R.: Description logics of minimal knowledge and negation as failure. ACM Trans. on Computational Logic **3**(2), 177–225 (2002). https://doi.org/10.1145/505372.505373, https://doi.org/10.1145/505372.505373
15. Franconi, E., Ibáñez-García, Y.A., Seylan, I.: Query answering with dboxes is hard. Electr. Notes Theor. Comput. Sci. **278**, 71–84 (2011). https://doi.org/10.1016/j.entcs.2011.10.007, https://doi.org/10.1016/j.entcs.2011.10.007
16. Gaggl, S.A., Rudolph, S., Schweizer, L.: Fixed-domain reasoning for description logics. In: Kaminka, G.A., Fox, M., Bouquet, P., Hüllermeier, E., Dignum, V., Dignum, F., van Harmelen, F. (eds.) Proc. of the 22nd Eur. Conf. on Artificial Intelligence (ECAI 2016). Frontiers in Artificial Intelligence and Applications, vol. 285, pp. 819–827. IOS Press (2016). https://doi.org/10.3233/978-1-61499-672-9-819, https://doi.org/10.3233/978-1-61499-672-9-819
17. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: A non-monotonic description logic for reasoning about typicality. Artif. Intell. **195**, 165–202 (2013)
18. Gogacz, T., Gutiérrez-Basulto, V., Ibáñez-García, Y.A., Murlak, F., Ortiz, M., Šimkus, M.: Ontology focusing: Knowledge-enriched databases on demand. In: Proc. 24th European Conf. on Artificial Intelligence (ECAI 2020) (to appear). IOS Press (2020), extended paper available at http://arxiv.org/abs/1904.00195
19. Gogacz, T., Lukumbuzya, S., Ortiz, M., Simkus, M.: Datalog rewritability and data complexity of $\mathcal{ALCHOIF}$ with closed predicates. In: Proc. Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2020). OIS Press (2020), to appear.
20. Gottlob, G., Kikot, S., Kontchakov, R., Podolskii, V.V., Schwentick, T., Zakharyaschev, M.: The price of query rewriting in ontology-based data access. Artif. Intell. **213**, 42–59 (2014). https://doi.org/10.1016/j.artint.2014.04.004, https://doi.org/10.1016/j.artint.2014.04.004
21. Knorr, M., Alferes, J.J., Hitzler, P.: Local closed world reasoning with description logics under the well-founded semantics. Artif. Intell. **175**(9-10), 1528–1554 (2011). https://doi.org/10.1016/j.artint.2011.01.007, http://dx.doi.org/10.1016/j.artint.2011.01.007
22. Levy, A.Y., Rousset, M.C.: Combining horn rules and description logics in {CARIN}. Artificial Intelligence **104**(1?2), 165 – 209 (1998)
23. Lukumbuzya, S., Ortiz, M., Simkus, M.: Resilient logic programs: Answer set programs challenged by ontologies. In: Proc. of the 34th AAAI Conf. on Artificial Intelligence (AAAI 2020). pp. 2917–2924. AAAI Press (2020), https://aaai.org/ojs/index.php/AAAI/article/view/5683
24. Lutz, C., Seylan, I., Wolter, F.: Ontology-based data access with closed predicates is inherently intractable(sometimes). In: Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI'2013). pp. 1024–1030. IJCAI/AAAI (2013)
25. Lutz, C., Seylan, I., Wolter, F.: Ontology-mediated queries with closed predicates. In: Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI 2015). pp. 3120–3126. AAAI Press (2015)
26. Lutz, C., Seylan, I., Wolter, F.: The data complexity of ontology-mediated queries with closed predicates. Logical Methods in Computer Science **15**(3) (2019). https://doi.org/10.23638/LMCS-15(3:23)2019, https://doi.org/10.23638/LMCS-15(3:23)2019
27. Motik, B., Rosati, R.: Reconciling description logics and rules. J. of the ACM **57**(5), 30:1–30:62 (2010). https://doi.org/10.1145/1754399.1754403, https://doi.org/10.1145/1754399.1754403

28. Ngo, N., Ortiz, M., Šimkus, M.: Closed predicates in description logics: Results on combined complexity. In: Proc. Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2016). pp. 237–246. AAAI Press (2016)
29. Xiao, G., Calvanese, D., Kontchakov, R., Lembo, D., Poggi, A., Rosati, R., Zakharyaschev, M.: Ontology-based data access: A survey. In: Lang, J. (ed.) Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI 2018). pp. 5511–5519. ijcai.org (2018). https://doi.org/10.24963/ijcai.2018/777, https://doi.org/10.24963/ijcai.2018/777