Ontology-Enriched Query Answering on Relational Databases

Shqiponja Ahmetaj[1*], Vasilis Efthymiou[2*], Ronald Fagin[3], Phokion G. Kolaitis[3, 4], Chuan Lei[3], Fatma Özcan[5*], Lucian Popa[3]

[1] TU Wien, Austria
[2] FORTH, Greece
[3] IBM Research - Almaden, USA
[4] UC Santa Cruz, USA
[5] Google, USA

ahmetaj@dbai.tuwien.ac.at, vefthym@ics.forth.gr, fagin@us.ibm.com, kolaitis@ucsc.edu, chuan.lei@ibm.com, fozcan@google.com, lpopa@us.ibm.com

Abstract

We develop a flexible, open-source framework for query answering on relational databases by adopting methods and techniques from the Semantic Web community and the data exchange community, and we apply this framework to a medical use case. We first deploy module-extraction techniques to derive a concise and relevant sub-ontology from an external reference ontology. We then use the chase procedure from the data exchange community to materialize a universal solution that can be subsequently used to answer queries on an enterprise medical database. Along the way, we identify a new class of well-behaved acyclic $\mathcal{EL}$-ontologies extended with role hierarchies, suitably restricted functional roles, and domain/range restrictions, which cover our use case. We show that such ontologies are C-stratified, which implies that the chase procedure terminates in polynomial time. We provide a detailed overview of our real-life application in the medical domain and demonstrate the benefits of this approach, such as discovering additional answers and formulating new queries.

Introduction

Ontology-Based Data Access (OBDA) (Xiao et al. 2018) and Ontology-Mediated Query Answering (OMQA) (Bienvenu 2016) are two closely related paradigms for answering queries over a database in the presence of an ontology that embodies semantic knowledge about the metadata. Much of the literature in OBDA/OMQA is devoted to the study of query answering under the assumptions that the ontology is available and the mappings between the database and the ontology have already been derived. There are real-life applications, however, where these assumptions need not hold.

A perusal of the query logs of a conversational system (Quamar et al. 2020) that we built for the medical database of IBM Micromedex®, used by medical experts (e.g., doctors, nurses, pharmacists) revealed that the answers obtained were incomplete and, furthermore, that many user queries could not be answered in the first place. This motivated enriching the medical database with an external reference ontology, such as SNOMED CT, so that the aforementioned shortcomings could be addressed.

When attempting to enrich query answers on a relational database by using an external ontology, two main challenges surface: (i) how to identify and reuse only the parts of the external ontology that are relevant to the given database - this is particularly acute when the external ontology is very large; and (ii) how to answer queries expressed over the vocabulary of the external ontology using the underlying database.

To address these challenges in a broader context, beyond our use case, we have developed a flexible open-source framework, whose design allows using different implementations for each component, with the aim of automating as much as possible an otherwise labor-intensive process.

For challenge (i) above, our framework uses existing tools from different AI communities, such as data management, Semantic Web, and description logics (DL), to automatically generate an ontology from a relational database (Lei et al. 2018), semi-automatically match this ontology with an external ontology (Knoblock et al. 2012; de Uña et al. 2018) and then extract, from the external ontology, a *module* that is relevant to the database (Grau et al. 2009).

For challenge (ii) above, we follow an approach based on data exchange, where we use the chase procedure to materialize a *universal solution*; this makes it possible to compute the *certain answers* of arbitrary conjunctive queries over the schema of the ontology by evaluating such queries on the materialized universal solution (Fagin et al. 2005). This approach bypasses the need to rewrite the queries (the *query-rewriting* approach) and fits the requirements of our target application, where we need to support ad-hoc queries that must be answered in real time. The underlying relational database is updated on only a monthly basis, hence the chase needs to be re-run on only a monthly basis as well. Note that query-rewriting approaches have two drawbacks that are relevant in our setting: each query has to be rewritten before it is evaluated, and the rewriting may be of exponential size in the size of the input query (Hernich et al. 2018). This may be inefficient when a large number of ad-hoc queries is asked.

We evaluate our framework on a use case involving an enterprise, medical relational database (MDB), enriched with the SNOMED CT medical ontology. It is well known that, for arbitrary ontologies, the chase procedure may not terminate. Motivated by our use case, we identify *acyclic*

*Work done while at IBM Research.

$\mathcal{ELH}^{fdr}$, a new class of ontologies that augment the $\mathcal{ELH}$ family of DLs (covering the expressivity of SNOMED CT) with limited functionality assertions, and with domain and range restrictions generated from MDB. We prove that acyclic $\mathcal{ELH}^{fdr}$ ontologies give rise to a C-stratified set of constraints (Meier, Schmidt, and Lausen 2009); therefore, for $\mathcal{ELH}^{fdr}$ ontologies, the chase always terminates.

We show that our framework makes it possible to not only obtain additional answers to queries over our medical database, but also to formulate and answer new queries that would not be meaningful without the external reference ontology (SNOMED CT). Furthermore, we demonstrate experimentally that the space and time overhead required by the chase approach is negligible compared to the benefits of obtaining additional query answers and answering more queries in real time.

In summary, the contributions of this work are:

- The adoption of AI and data exchange methods and tools in a real-world medical use case, with the aim of expanding the capabilities of an existing product.
- Backing our use case by concrete theoretical guarantees; we define acyclic $\mathcal{ELH}^{fdr}$, and show it is C-stratified, which implies that the standard chase always terminates in polynomial time.
- A reference architecture for an end-to-end framework for ontology-enriched query answering, generalizing the applicability of our targeted use case. Our framework is publicly available on github[1], and it can be deployed on top of existing question-answering and conversational systems.
- An experimental evaluation showing the benefits of employing AI and data exchange technologies for real-world industrial problems: we get more query answers by exploiting SNOMED CT as an external reference ontology.

Preliminaries

Data Exchange A *relational schema* S is a collection of relation symbols of specified arities. A relational database *instance* $\mathcal{I}$ over S is a collection of relations whose arities match those of the relation symbols in S. The *domain* of $\mathcal{I}$ is the set of all values that occur in the relations of $\mathcal{I}$. All instances are assumed to be *finite*, i.e., they consist of finite relations. An *atom* (over S) is a formula $P(x_1, \ldots, x_m)$, where P is a relation symbol in S and $x_1, \ldots, x_m$ are variables, not necessarily distinct. A *fact* of an instance $\mathcal{I}$ over S is an expression $P^{\mathcal{I}}(a_1, \ldots, a_m)$, where P is a relation symbol in R, where $P^{\mathcal{I}}$ is the relation in $\mathcal{I}$ interpreting P, and $a_1, \ldots, a_m$ are values such that $(a_1, \ldots, a_m) \in P^{\mathcal{I}}$.

A *data exchange setting* (also known as a *schema mapping*) is a quadruple $\mathcal{M} = (S, T, \Sigma_{st}, \Sigma_t)$, where S is a source schema, T is a target schema, Σ_{st} is a finite set of source-to-target constraints, and Σ_t is a finite set of constraints over the target schema (Fagin et al. 2005). The constraints that we consider in Σ_{st} and Σ_t are as follows. First, a *tuple-generating dependency (tgd)* is a constraint α of the form $\forall \mathbf{x} \forall \mathbf{y} (\phi(\mathbf{x}, \mathbf{y}) \to \exists \mathbf{z} \psi(\mathbf{x}, \mathbf{z}))$, where ϕ and ψ are conjunctions of atoms and every variable in $\mathbf{x}$ occurs in both ϕ and ψ. We may call $\phi(\mathbf{x}, \mathbf{y})$ the *body* of the tgd and $\exists \mathbf{z} \psi(\mathbf{x}, \mathbf{z})$ the *head* of the tgd. We will often omit writing the universal quantifiers $\forall \mathbf{x} \forall \mathbf{y}$. In a data exchange setting, the set Σ_{st} is a set of *source-to-target tgds (st-tgds)*, i.e., tgds α such that ϕ consists of atoms from the source schema S, and ψ consists of atoms from the target schema T. The set Σ_t consists of *target tgds (t-tgds)*, i.e., tgds defined entirely in terms of the target schema T, along with *target equality-generating dependencies (t-egds)*, i.e., constraints of the form $\forall \mathbf{x} (\phi(\mathbf{x}) \to x_1 = x_2)$, where ϕ consists of atoms from T and the variables x_1 and x_2 occur in $\mathbf{x}$.

A target instance $\mathcal{J}$ is a *solution* for a source instance $\mathcal{I}$ w.r.t. $\mathcal{M}$ if the pair $(\mathcal{I}, \mathcal{J})$ satisfies $\mathcal{M}$, i.e., $\mathcal{I}$ and $\mathcal{J}$ together satisfy every constraint in Σ_{st}, and $\mathcal{J}$ satisfies every constraint in Σ_t. Fix a data exchange setting $\mathcal{M}$. The *chase* is an algorithm that, given a source instance $\mathcal{I}$ as input, computes a special solution (called a *universal solution*) $\mathcal{J}$ for $\mathcal{I}$ w.r.t. $\mathcal{M}$ (Fagin et al. 2005). Intuitively, a universal solution is a "most general" solution, because it can be homomorphically mapped into every other solution; thus, universal solutions are the preferred solutions to materialize in data exchange. There are several different variants of the chase; here we will only refer to the *standard chase* (also known as *non-oblivious chase*).

Certain Answers A conjunctive query q is a first-order formula $\exists \bar{y} \phi(\bar{x}, \bar{y})$, where $\phi(\bar{x}, \bar{y})$ is a conjunction of atoms, where $\bar{y}$ consists only of variables, and $\bar{x}$ consists of constants or variables. If q is a single atom without existentially quantified variables, i.e., an atom of the form $P(\vec{x})$, then it is called an *instance query*. If q is a query over the target schema T, and $\mathcal{I}$ is a source instance, then the *certain answers* $cert(q, \mathcal{I}, \mathcal{M})$ of q on $\mathcal{I}$ w.r.t. a data exchange setting $\mathcal{M} = (S, T, \Sigma_{st}, \Sigma_t)$ are defined as

$$cert(q, \mathcal{I}, \mathcal{M}) = \bigcap \{q(\mathcal{J}) : \mathcal{J} \text{ is a solution for } \mathcal{I} \text{ w.r.t. } \mathcal{M}\}.$$

As shown in (Fagin et al. 2005), if $\mathcal{J}$ is a universal solution for $\mathcal{I}$ w.r.t. $\mathcal{M}$, then the certain answers of every conjunctive query q over T can be obtained by evaluating q on $\mathcal{J}$ and then removing all tuples containing null values (null values can arise during the chase).

$\mathcal{ELH}$ terminologies Let $\mathsf{N_R}$ and $\mathsf{N_C}$ be countably infinite, mutually disjoint sets of *role names* (binary relations) and *concept names* (unary relations), respectively. *Concepts* are built as follows (see (Baader et al. 2017) for more details):

$$C := A \mid \top \mid \exists r.C \mid C \sqcap D,$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and C and D are concepts. Intuitively, concepts represent sets of objects, while roles represent binary relations between objects. In particular, $\exists r.C$ denotes the objects that are related through a role name r to some object that is an instance of the concept C, and the concept $C \sqcap D$ denotes the objects that are instances of the concepts C and D. In OWL, concepts are called classes and roles are called properties.

[1]https://github.com/IBM/ontology-enriched-query-answering

An $\mathcal{ELH}$ *terminology* $\mathcal{T}$ (a.k.a. *restricted TBox*) is a set of *concept definitions* $A \equiv C$, *(primitive) concept inclusions* $A \sqsubseteq C$, and *role inclusions* $r \sqsubseteq s$, where $A \in \mathsf{N_C}$, C is a concept different from $\top$, $\top \sqcap \top$, etc., and where r, s are role names. Note that $A \equiv C$ is an abbreviation for $A \sqsubseteq C$ and $C \sqsubseteq A$. As usual, we denote with $\sqsubseteq^*_{\mathcal{T}}$ the smallest relation that satisfies: (1) $r \sqsubseteq^*_{\mathcal{T}} r$ for every role name r in $\mathcal{T}$, and (2) if $r \sqsubseteq^*_{\mathcal{T}} r_1$ and $r_1 \sqsubseteq s \in \mathcal{T}$, then $r \sqsubseteq^*_{\mathcal{T}} s$. This binary relation between role names appearing in $\mathcal{T}$ captures all relevant role inclusions implied by $\mathcal{T}$.

In the following, we mention general (not restricted) TBoxes, which intuitively, extend terminologies by allowing for *general* concept inclusions of the form $C \sqsubseteq D$, where C and D are arbitrary concepts. Note that such axioms are not allowed in terminologies, where only concept names can occur on the left-hand side of concept definitions and primitive concept inclusions. We use the term *signature* to refer to a set of concept and role names, while the *signature of* $\mathcal{T}$, denoted $sig(\mathcal{T})$, refers to the set of concept and role names appearing in a DL TBox $\mathcal{T}$.

Framework Architecture

In this section, we present the architecture of our framework, as shown in Figure 1. In this figure, rectangles in the dashed boxes correspond to components of our framework and edges correspond to data flow. The numbering of the components (in circles) illustrates the order of execution.

Step 1: Ontology Creation. First, we generate a TBox $\mathcal{T}_1$ extracted from the relational schema S. For this component, we follow the approach suggested in (Lei et al. 2018), which, unlike other approaches generating so-called "flat" ontologies (Calvanese et al. 2017; Rodriguez-Muro, Kontchakov, and Zakharyaschev 2013; Efthymiou et al. 2017), enables the inference of concept hierarchies, domain and range restrictions for roles, as well as functional dependencies. In brief, for tables in the relational schema, we first identify primary and foreign keys. Then, we consider tables with exactly two columns, both acting as foreign keys to different tables in the schema, as intermediate tables.

From every non-intermediate table R, we generate a concept containing the primary key column of R. Subsequently, from every column x_i, with $1 < i \leq n$, in a table $R(\underline{x_1}, \ldots, x_n)$, we generate a role $r_{x_i}(A, B)$, whose domain A is restricted to values from x_1, where x_1 is the (non-composite) primary key of R, and range B is either restricted to values of a specific data type (e.g., string, integer, date), or to values from a column x'_j of another table R', if x_i is a foreign key referring to R'. If there is a table R_1 with a single column, which is a foreign key referring to a table R_2, we consider the concept C_1 generated from R_1 as subsumed by the concept C_2 generated from R_2. We consider the remaining roles generated from non-intermediate tables to be functional roles. Intuitively, a functional role is one that cannot associate the same instance to two different instances. Examples of functional roles can be "hasBiologicalMother", associating a person to his/her biological mother (since a person cannot have more than one biological mother), "capitalOf", associating a capital city to a country, and "hasSSN" associating a person to his/her social security number.

Example. Consider a relational schema with the tables/relations Drug(drugId, drugName, drugClass) and DrugClass(drugClassId, drugClassName), whose primary keys are underlined, while drugClass is a foreign key referring to the primary key drugClassId of the relation DrugClass. The ontology creation step will first generate two concepts named Drug and $\mathsf{DrugClass}$, since none of the two tables is an intermediate table. From the Drug table, we will also generate the roles (or OWL properties) $\mathsf{drugName}$ and $\mathsf{drugClass}$, both having as domain the instances of the concept Drug. The range of $\mathsf{drugName}$ will be restricted to string values. In OWL, these types of roles correspond to the so-called datatype properties. The range of the $\mathsf{drugClass}$ concept will be the instances of the concept $\mathsf{DrugClass}$. From the DrugClass table, we will generate the role $\mathsf{drugClassName}$, whose domain will be the instances of the $\mathsf{DrugClass}$ concept, and whose range will be restricted to strings. Since the given tables are not intermediate, all the three generated roles will be functional.

Step 2: Matchings Generation. Given the TBox $\mathcal{T}_1$, and the external TBox $\mathcal{T}_2$, we identify concepts and roles from $\mathcal{T}_1$ that correspond to concepts and roles in $\mathcal{T}_2$, and return those matchings as a set $\mathcal{C} = \{(N_1, N_2)\}$ of pairs, where $N_i \in sig(\mathcal{T}_i)$, $i \in \{1, 2\}$.

As a follow-up to the example of Step 1, let $\mathcal{T}_1$ be the ontology generated at that step and assume that the external TBox $\mathcal{T}_2$ contains a concept called $\mathsf{Medicament}$, which matches with the concept Drug from $\mathcal{T}_1$. Intuitively, this means that $\mathsf{Medicament}$ and Drug represent information about the same concept. Thus, the set of matchings will contain the elements $(\mathsf{Drug}, \mathsf{Medicament})$.

For this component, we can either rely on fully automated methods (e.g., LogMap (Jiménez-Ruiz and Grau 2011)), or use semi-automatic methods (e.g., SERENE (de Uña et al. 2018), KARMA (Knoblock et al. 2012)). Semi-automatic methods suggest some initial matchings, and those matchings are then modified by domain experts. Note that this is the only step in which human intervention may be needed.

We would like to remark on the importance of this step for the whole framework. If the generated matchings are not complete, this will result in generating incomplete query answers. Worse yet, if the matchings are not correct, this will also affect the correctness of the answers provided by our framework, and this may be critical for some domains, such as medicine. Thus, we prefer the option of semi-automatic matching, in which human experts can manually inspect the process and guarantee the correctness of the results.

Step 3: Module extraction. Since the external ontology $\mathcal{T}_2$ may contain a very large number of axioms, and a great part of these axioms may not be relevant to $\mathcal{T}_1$, we want to retain a small subset $\mathcal{T}'_2$ of $\mathcal{T}_2$ that captures only the meaning of the terms in a signature $\mathbb{S}$ that we are interested in. This is known in the literature as extracting a minimal $\mathbb{S}$-module in $\mathcal{T}_2$ (Grau et al. 2008). Intuitively, the minimal $\mathbb{S}$-module extracted from $\mathcal{T}_2$ would give us the same answers as if querying the whole ontology $\mathcal{T}_2$ when posing queries over the given signature $\mathbb{S}$. It has been proven that this is an $\mathsf{ExpTime}$-complete problem, even for the less expressive

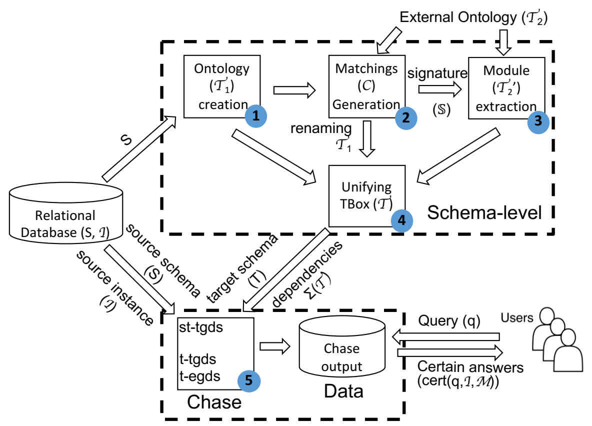
External Ontology ($\mathcal{T}_2'$)
Ontology ($\mathcal{T}_1'$) creation
1
Matchings ($\mathcal{C}$) Generation
2
signature ($\mathbb{S}$)
Module ($\mathcal{T}_2'$') extraction
3
renaming $\mathcal{T}_1$
S
Unifying TBox ($\mathcal{T}$)
4
Schema-level
Relational Database (S, $\mathcal{I}$)
source schema (S)
source instance ($\mathcal{I}$)
target schema (T)
dependencies $\Sigma(\mathcal{T})$
st-tgds
t-tgds
t-egds
5
Chase
Chase output
Data
Query (q)
Users
Certain answers (cert(q,$\mathcal{I}$,$\mathcal{M}$))

Figure 1: Framework architecture.

fragments of DL, and undecidable for the more expressive fragments (Grau et al. 2008). Therefore, we leverage an efficient $\mathbb{S}$-module extraction algorithm that is based on the notion of $\bot\top^*$-syntactic locality (Grau et al. 2009). We include in $\mathbb{S}$ all the names from $\mathcal{T}_2$ that appear in the matchings: $\mathbb{S} = \{N_2 | (N_1, N_2) \in \mathcal{C}\}$.

Example. Continuing our running example, assume that the external TBox $\mathcal{T}_2$, in addition to the concept $\mathsf{Medicament}$, contains also the concepts Cell and Zygote, and assume that no relation between these concepts and the concept $\mathsf{Medicament}$ can be inferred from $\mathcal{T}_2$. Considering the set of matchings found in Step 2, namely $\mathcal{C} = \{(\mathsf{Drug}, \mathsf{Medicament})\}$, we are interested to extract from (the possibly large) $\mathcal{T}_2$ a substructure that contains all the relevant knowledge about the concept $\mathsf{Medicament}$. To this aim, in the module extraction step, we will define the signature $\mathbb{S}$ to be $\{\mathsf{Medicament}\}$ and we will extract an $\mathbb{S}$-module $\mathcal{T}_2'$. Since the concepts Cell and Zygote do not contribute in the knowledge about the concept $\{\mathsf{Medicament}\}$, they are not expected to appear in the extracted module $\mathcal{T}_2'$.

The benefits of module extraction became evident in our empirical study, where using the whole $\mathcal{T}_2$ instead of its $\mathbb{S}$-module $\mathcal{T}_2'$ resulted in a 300,000% increase (from 154 to 466,627) in the number of t-tgds, and the chase had not yet terminated after 12 hrs, while when using $\mathcal{T}_2'$, it terminated in less than 2 sec. The overhead of this step was 20 sec, out of which 14 were required to load SNOMED CT in memory.

Step 4: Unifying the TBox. In this step, we perform a renaming operation and a union operation. First, for every matching pair $(N_1, N_2) \in \mathcal{C}$, we replace every occurrence of N_1 in $\mathcal{T}_1$ with its matched name N_2. Finally, we return the unified TBox $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2'$. In our running example, this unification step would result in a single ontology with the concepts $\mathsf{Medicament}$ and $\mathsf{DrugClass}$. The roles $\mathsf{drugName}$ and $\mathsf{drugClass}$, whose domain previously was bound to instances of the concept Drug, will now have as their domain instances of the concept $\mathsf{Medicament}$, while their range will remain unchanged. The role $\mathsf{drugClassName}$ will not be affected by this unification, and it will be also part of the unified ontology.

Step 5: Query Answering via the Chase. Our data exchange setting $\mathcal{M}$ uses the relational schema S as the source schema, a schema T resulting from the unified TBox $\mathcal{T}$ (from Step 4) as the target schema, and a set of s-t tgds, t-tgds and t-egds that we generate as we shall discuss. After running the chase with the setting $\mathcal{M}$ on the source instance $\mathcal{I}$, we can then compute certain answers $cert(q, \mathcal{I}, \mathcal{M})$ for queries q expressed over T. The details of this step are provided next.

Based on the results reported on a recent benchmarking effort (Benedikt et al. 2017) and on the features that publicly available chase implementations support (e.g., t-egds, query answering), we determined that RDFox (Nenov et al. 2015) is the most suitable implementation for our purposes.

Chase Termination for Acyclic $\mathcal{ELH}^{fdr}$

In our use case, we are interested in $\mathcal{ELH}$ terminologies extended with domain and range restrictions of the form $\mathsf{dom}(r) \sqsubseteq A$ and $\mathsf{rng}(r) \sqsubseteq A$ (i.e., the domain or the range of the role r is contained in the concept name A), and functionality assertions $\mathsf{func}(r)$. This DL fragment covers our unified TBox (after Step 4), since SNOMED CT is acyclic $\mathcal{ELH}$ (i.e., without domain and range restrictions, or functional dependencies) and since the ontology generated from our database, after Step 1, falls under acyclic $\mathcal{EL}$ with domain and range restrictions, as well functional assertions. At the same time, we want such extended terminologies to have the property that the chase always terminates. There is a well known notion of an *acyclic* $\mathcal{EL}$ terminology for which the chase always terminates (e.g., see Definition 2.9 in (Baader et al. 2017)). The addition of domain/range restrictions and functionality assertions, however, may destroy the termination of the chase (e.g., consider the axioms $A \sqsubseteq \exists r$ and $\mathsf{rng}(r) \sqsubseteq A$). For this reason, we introduce a new notion of acyclicity for $\mathcal{ELH}$ terminologies extended with domain and range restrictions, and restricted functionality assertions.

Definition 1. *An* $\mathcal{ELH}^{fdr}$ *terminology is an* $\mathcal{ELH}$ *terminology* $\mathcal{T}$ *extended with* domain restrictions $\mathsf{dom}(r) \sqsubseteq A$, range restrictions $\mathsf{rng}(r) \sqsubseteq A$, *where* A *is a concept name, and* functionality assertions $\mathsf{func}(r)$ *that satisfy the following:*

- *if* $r \sqsubseteq s \in \mathcal{T}$ *with* $r \neq s$, *then* $\mathsf{func}(s) \notin \mathcal{T}$, *and*
- *if* r *appears in* C *and* $A \sqsubseteq C \in \mathcal{T}$ *or* $A \equiv C \in \mathcal{T}$, *then* $\mathsf{func}(r) \notin \mathcal{T}$.

The semantics of these is standard (Baader et al. 2017).

Definition 2. *Let* $\mathcal{T}$ *be an* $\mathcal{ELH}^{fdr}$ *terminology.*

- *We say that a concept name* A directly uses *a concept name* B *if there is a concept inclusion* $A \sqsubseteq C \in \mathcal{T}$ *or a concept definition* $A \equiv C \in \mathcal{T}$, *where* C *is a possibly complex concept, such that at least one of the following holds:*

1. *B occurs in the expression defining C, or*
2. *there exists a role name r in C such that either* $\mathsf{rng}(s) \sqsubseteq B \in \mathcal{T}$ *and* $r \sqsubseteq^*_{\mathcal{T}} s$, *or* $\mathsf{dom}(s) \sqsubseteq B \in \mathcal{T}$ *and* $r \sqsubseteq^*_{\mathcal{T}} s$.

- *We say that a concept name* A uses *a concept name* B *if either* A *directly uses* B, *or there is a concept name* B' *such that* A *uses* B', *and* B' *directly uses* B.
- *We say that* $\mathcal{T}$ *is* acyclic *if (a) there is no concept name in* $\mathcal{T}$ *that uses itself, and (b) no concept name occurs more*

than once on the left-hand side of a concept inclusion $A \sqsubseteq C \in \mathcal{T}$ or a concept definition $A \equiv C \in \mathcal{T}$.

It is well known and easy to see that for every $\mathcal{EL}$-concept C, there is a conjunctive query $q_C(x)$ with a free variable x, such that $C(x) \equiv q_C(x)$. We can further distinguish two cases for such a conjunctive query $q_C(x)$. **Case 1.** $q_C(x)$ has existential quantifiers that bind at least one variable in the conjunction, i.e., it is of the form $q_C(x) := \exists \bar{y} \phi_C(\bar{y}, x)$, where $\bar{y}$ is a non-empty tuple of variables. **Case 2.** $q_C(x)$ has no existential quantifiers, i.e., it is of the form $q_C(x) := A_1(x) \wedge \ldots \wedge A_n(x)$, where $A_1, \ldots, A_n$ are concept names.

Data Exchange Setting We now describe the data exchange setting of our case study.

Source and target schemas. As a source schema, we use the schema S of the input relational database (see Figure 1). We generate the target schema T from the unified TBox $\mathcal{T}$ by viewing concepts as unary relations and roles as binary relations. At the chase implementation level, we follow the so-called "common" data format (Benedikt et al. 2017).

st-tgds. We transform every relation R of arity $n > 2$ in the source schema S to unary and binary relations by generating the following tgd:

$$R(\underline{x_1}, \ldots, x_n) \rightarrow R'(x_1) \wedge R'^{1,2}(x_1, x_2) \wedge \ldots \wedge R'^{1,n}(x_1, x_n),$$

where x_1 is assumed to be used as the (non-composite) primary key of R in S, and R', $R'^{1,j}$ are fresh relation names. If $(R, R'') \in \mathcal{C}$, we replace $R'(x_1)$ above with $R''(x_1)$. Even if $\mathcal{C}$ contains names from $\mathcal{T}_1$ and $\mathcal{T}_2$ (as defined in Step 2 of Framework Architecture), the names used in $\mathcal{T}_1$ are by convention the same as those used in the schema S.

t-tgds. All axioms other than the functional roles of an $\mathcal{ELH}^{fdr}$ terminology can be expressed as tgds. For this, we treat each axiom $A \equiv C$ as the two inclusions $A \sqsubseteq C$ and $C \sqsubseteq A$; we also use the aforementioned fact that every $\mathcal{EL}$-concept is defined by a conjunctive query. The tgds arising from an $\mathcal{ELH}^{fdr}$ terminology have one of the following seven types.

$$A(x) \rightarrow \exists \bar{y} \phi_C(\bar{y}, x) \quad \text{(arises from } A \sqsubseteq C, \text{ where } C \text{ is of Case 1)} \quad (1)$$

$$A(x) \rightarrow A_1(x) \wedge \ldots \wedge A_n(x) \quad \text{(arises from } A \sqsubseteq C, \text{ where } C \text{ is of Case 2)} \quad (2)$$

$$\phi_C(\bar{y}, x) \rightarrow A(x) \quad \text{(arises from } C \sqsubseteq A, \text{ where } C \text{ is of Case 1)} \quad (3)$$

$$A_1(x) \wedge \ldots \wedge A_n(x) \rightarrow A(x) \quad \text{(arises from } C \sqsubseteq A, \text{ where } C \text{ is of Case 2)} \quad (4)$$

$$r_1(x, y) \rightarrow r_2(x, y) \quad \text{(arises from } r_1 \sqsubseteq r_2) \quad (5)$$

$$r(x, y) \rightarrow A(x) \quad \text{(arises from } \mathsf{dom}(r) \sqsubseteq A) \quad (6)$$

$$r(x, y) \rightarrow A(y) \quad \text{(arises from } \mathsf{rng}(r) \sqsubseteq A) \quad (7)$$

t-egds. Every functional role r gives rise to the egd $r(x, y) \wedge r(x, z) \rightarrow y = z$.

If $\mathcal{T}$ is an $\mathcal{ELH}^{fdr}$ terminology, then we write $\Sigma(\mathcal{T})$ for the set of t-tgds and t-egds associated with $\mathcal{T}$ as above.

Chase Termination One may think that the tgds arising from an acyclic $\mathcal{EL}$ terminology form a *weakly acyclic* set, guaranteeing chase termination (Fagin et al. 2005). This, however, is not true. Indeed, let $\mathcal{T}$ be the $\mathcal{EL}$ terminology consisting of the two axioms $A \sqsubseteq \exists r$ and $B \equiv \exists r.A$. Clearly, $\mathcal{T}$ is acyclic, but one can check that the set $\Sigma(\mathcal{T})$ of the three tgds arising from these two axioms is *not* weakly acyclic. In fact, since it contains no egds, $\Sigma(\mathcal{T})$ is *not* even *super-weakly acyclic* (Marnette 2009). For this reason, we need to consider a generalization of weak acyclicity, called C-*stratification* (Meier, Schmidt, and Lausen 2009).

Recall that in the case of the standard chase, a tgd α of the form $\forall \mathbf{x} \forall \mathbf{y} (\phi(\mathbf{x}, \mathbf{y}) \rightarrow \exists \mathbf{z} \psi(\mathbf{x}, \mathbf{z}))$ is applicable on an instance $\mathcal{J}$ if there are tuples $\mathbf{a}$ and $\mathbf{b}$ of values from the domain of $\mathcal{J}$ such that $\mathcal{I} \models \phi(\mathbf{a}, \mathbf{b})$, but there is no tuple $\mathbf{c}$ such that $\mathcal{J} \models \psi(\mathbf{a}, \mathbf{c})$. In the variant of the chase known as the *oblivious chase*, such a tgd is applicable on $\mathcal{J}$ if, simply, there are tuples of values $\mathbf{a}$ and $\mathbf{b}$ such that $\mathcal{I} \models \phi(\mathbf{a}, \mathbf{b})$. Moreover, we say that an instance $\mathcal{K}$ is obtained from $\mathcal{J}$ in a *single step* of the oblivious chase if $\mathcal{K}$ consists of the facts of $\mathcal{J}$ together with facts involving fresh nulls $\mathbf{u}$ that interpret the variables $\mathbf{z}$ so that $\mathcal{K} \models \psi(\mathbf{a}, \mathbf{u})$. The definition of an egd being applicable in the oblivious chase is similar.

Definition 3. *Let α and β be two tgds. We say that α precedes β, denoted $\alpha \prec \beta$, if there are instances $\mathcal{J}$ and $\mathcal{K}$ such that the following hold:*

- *$\mathcal{K}$ is obtained from $\mathcal{J}$ in a single step of the oblivious chase using α.*
- *There are tuples $\mathbf{a}'$, $\mathbf{b}'$ of values from the domain of $\mathcal{K}$ such that $\mathcal{J} \models \phi'(\mathbf{a}', \mathbf{b}') \rightarrow \exists \mathbf{z}' \psi'(\mathbf{a}', \mathbf{z}')$, but $\mathcal{K} \not\models \phi'(\mathbf{a}', \mathbf{b}') \rightarrow \exists \mathbf{z}' \psi'(\mathbf{a}', \mathbf{z}')$, where we assume that the tgd β is $\forall \mathbf{x}' \forall \mathbf{y}' (\phi'(\mathbf{x}', \mathbf{y}') \rightarrow \exists \mathbf{z}' \psi'(\mathbf{x}', \mathbf{z}'))$.*

In a similar way, we define the notion $\alpha \prec \beta$, where α and β are egds.

Definition 4. *Let Σ be a set of tgds and egds.*

- *The* chase graph *of Σ is the graph with nodes the elements of Σ and edges (α, β) such that $\alpha \prec \beta$.*
- *Σ is* C-stratified *if for every cycle $\mathcal{D}$ of the chase graph of Σ, the tgds in $\mathcal{D}$ form a weakly acyclic set.*

In (Meier, Schmidt, and Lausen 2009), it was shown that if a set Σ of tgds and egds is C-stratified, then, on every input database instance $\mathcal{J}$, the standard chase w.r.t. Σ terminates in time bounded by a polynomial in the size of $\mathcal{J}$. Next, we show that every acyclic $\mathcal{ELH}^{fdr}$ terminology is C-stratified.

Theorem 1. *Let $\mathcal{T}$ be an acyclic $\mathcal{ELH}^{fdr}$ terminology and let $\Sigma(\mathcal{T})$ be the associated set of tgds and egds. Then $\Sigma(\mathcal{T})$ is C-stratified.*

Proof. (*Hint*) The proof entails a delicate analysis of the structure of cycles in the chase graph of $\Sigma(\mathcal{T})$. Specifically, the following two claims are established for an arbitrary cycle $\mathcal{D}$ in the chase graph consisting of tgds and egds.

Claim 1. The cycle $\mathcal{D}$ consists entirely of tgds; moreover, none of the tgds in the cycle $\mathcal{D}$ is of Type 2 or of Type 4.

Claim 2. If one of the tgds in the cycle $\mathcal{D}$ has existentially quantified variables in its head, then it is of the form $A(x) \rightarrow \exists \bar{y} \phi_C(\bar{y}, x)$; furthermore, the concept name A uses every concept name that occurs in another tgd in the cycle.

Claim 2 implies that if one of the tgds in the cycle $\mathcal{D}$ has existentially quantified variables in its head, then $\mathcal{T}$ contains a concept name A that uses itself, which contradicts the hypothesis that $\mathcal{T}$ is an acyclic $\mathcal{ELH}^{fdr}$ terminology. Thus, every cycle in the chase graph of $\Sigma(\mathcal{T})$ consists of full tgds only; this implies that the set of the tgds of every such cycle forms a weakly acyclic set; hence $\Sigma(\mathcal{T})$ is C-stratified. □

Evaluation

In this section, we present the experimental evaluation of our framework for our use case, involving an enterprise, medical relational database, which we will call MDB, and SNOMED CT as the external ontology. The evaluation is carried out at two levels and demonstrates the benefits of using our framework: first, we document the discovery of additional answers for queries that were originally asked directly on MDB; second, we document the ability to formulate new queries, when using our framework, that were not meaningful using only the schema of MDB.

Setup. The experiments reported here were executed on a laptop running macOS® 10.15.2 with a Quad-Core Intel® i7@2.9GHz processor and 16 GB RAM. The times reported are the average from 10 executions.

Data. MDB is an enterprise relational database, consisting of 62 relations of arities varying from 2 to 11, and of 158 foreign key constraints with a total of 512,769 tuples, occupying 62.3 MB disk space. MDB data, related to drug information, such as dosage and drug interactions, support the conversation system (Quamar et al. 2020) that motivated this work. The relational schema and instance of MDB are updated on a yearly and monthly basis, respectively.

Ontologies. The ontology $\mathcal{T}_1$, which is generated automatically from MDB using the methodology of (Lei et al. 2018), consists of 49 concepts and 170 roles. All 170 roles are associated with both domain and range restrictions, while 156 of them are functional. The ontology $\mathcal{T}_2$ is SNOMED CT, and consists of 356,065 concepts and 119 roles. None of the roles in SNOMED CT is functional, or associated with domain or range restrictions. We identified 12 matchings between $\mathcal{T}_1$ and $\mathcal{T}_2$, i.e., $|\mathcal{C}| = 12$. As a side note, we observe that there are concepts in SNOMED CT which appeared as entries of tuples in our MDB tables. This was another challenge for matching, a challenge that is not usually addressed in the literature, but it is rather frequent in real applications. The module $\mathcal{T}_2'$ for the signature $\mathbb{S}$, as defined in Framework Architecture, in $\mathcal{T}_2$ consists of 35 concepts and 7 roles. Finally, the unified TBox $\mathcal{T}$ consists of 72 concepts and 177 roles. Out of these roles, 156 are functional, while 170 are associated with domain and range restrictions. The small number of concepts in $\mathcal{T}$, as compared to those in $\mathcal{T}_2$, is the result of module extraction; it makes the chase process much more efficient in terms of time and space requirements, as described shortly.

Given that SNOMED CT is a standard ontology, using standardized terminology, it is intuitive for medical experts (e.g., pharmacists, doctors, and nurses), i.e., our target group of users, to understand and use the unified ontology without much effort.

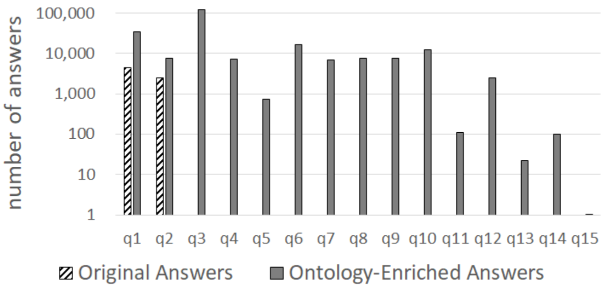

Figure 2: Number of query answers (in log scale).

Chase. Our data exchange setting has 62 st-tgds, 154 t-tgds, and 156 t-egds. The chase terminated successfully in 1,676 ms, which breaks into 870 ms for applying the st-tgds, and 806 ms for applying the t-tgds and t-egds. The result occupied 77.5 MB of space, which is a 24% increase compared to the 62.3 MB required to store the source instance.

Results. Figure 2 presents the evaluation results for a subset of real queries posed by medical experts on MDB, from the query logs of a conversation system (Quamar et al. 2020) from January to June 2019. The figure contains the number of answers returned by running only the st-tgds (Original Answers), i.e., renaming MDB names to their matched SNOMED names, and the number of answers returned by using our framework (Ontology-Enriched Answers). The evaluated queries, expressed in the vocabulary of the target schema are (written in Prolog style; quantifiers omitted):

$q_1(x) :=$ ClinicalFinding(x) .
$q_2(x) :=$ DrugAdministration(x) .
$q_3(x,y,z) :=$ ClinicalFinding(x), DrugInteraction(x), dIForDrug(x,w), drugName(w,y), dISeverity(x,z) .
$q_4(x,y) :=$ ClinicalFinding(x), DrugInteraction(x), dISeverity(x,y), dIForDrug(x,w), drugName$(w,$ "Aspirin"$)$.
$q_5(y,s) :=$ ClinicalFinding(x), dIForDrug(x,w), drugName$(w,$ "Aspirin"$)$, dISeverity$(x,$ "Major"$)$, dIForDrug(x,z), drugName(z,y), dISummary(x,s).
$q_6(x) :=$ Disease(x), roleGroup(x,y), associatedWith(y,z), DrugOrMedicament(z).
$q_7(x) :=$ ClinicalHistoryAndObservationFindings(x) .
$q_8(x) :=$ ManagementOfDrugAdministration(x) .
$q_9(x) :=$ ManagementProcedure(x) .
$q_{10}(x) :=$ PreventionStatus(x) .
$q_{11}(x) :=$ Substance(x) .
$q_{12}(x,z) :=$ MedicinalProduct(x), playsRole(x,y), TherapeuticRole(y), drugClassName(x,z).
$q_{13}(x) :=$ PreventionStatus(x), monitoringOfDrug(x,y), drugName$(y,$ Ibuprofen"$)$.
$q_{14}(x) :=$ drugName(z,x), Complication(y), advEffectDescription$(y,$ "Seizure"$)$, advEffectToDrug(y,z).
$q_{15}(x) :=$ Procedure(x), drugRouteType$(y,$ "Oral"$)$, procedRoute(x,y), procedToDrug(x,z), drugName$(z,$ "Aspirin"$)$.

Queries q_1 and q_2 are instance queries that MDB could answer, but incompletely. Query q_3 is a CQ that MDB could answer, but provided 0 answers, because MDB does not know that Drug Interaction is a Clinical Finding. Queries q_4 and q_5 are similar to q_3, but they include constant values, i.e., they are specifically asking for a certain drug ("Aspirin"), as well as drugs whose interactions with Aspirin have a certain severity ("Major"). Query q_6 is a CQ that MDB could not answer, as it consists of concept and role names only met in SNOMED. Queries q_7-q_{11} (resp. q_{12}-q_{15}) are instance queries (resp. CQs) that MDB could not answer, as they include some concepts and roles from SNOMED, not included in MDB. Query-answering times ranged from 1ms (for queries q_{13}-q_{15}) to 576ms (for q_3), averaging 64ms.

We identify two general types of queries, for which our framework is beneficial: (i) Queries whose conjuncts are all known in the relational database, but where we learned something new about those conjuncts from the external ontology (e.g., their involvement in a new concept/role hierarchy). Examples of such queries are q_1-q_5. (ii) Queries whose conjuncts include concept/role names unknown to the relational database, but known to the external ontology; this is the reason that, for such queries, there are no answers. Examples of such queries are q_6-q_{15}.

In summary, we showed that our framework offers ontology-enriched answers to real queries from the query log of a conversational system in production. Note that this system was built with the queries that a relational database can answer. The log revealed that medical experts were also asking natural language questions using vocabulary not captured by the database, but was available from SNOMED CT. By enriching the database, we were able to answer those queries, as well as provide additional answers to existing ones. Since the answers that our framework provides are certain answers, this guarantees their correctness, provided that the produced matchings are also correct. This significantly expands the scope and increases the usability of the conversational system.

Related Work

The OBDA paradigm has received increasing attention in the last decade, giving rise to a large body of applications (Xiao et al. 2018; Giacomo et al. 2018). Typical OBDA systems, e.g, Mastro (Calvanese et al. 2011), Ontop (Calvanese et al. 2017), Optique (Kharlamov et al. 2015), cannot be directly used for our case study, since they target OWL2 QL ontologies, which are based on DL-Lite, a DL that is orthogonal to $\mathcal{ELH}$. A major difference with (Hovland et al. 2017), which manually defines an ontology that captures the vocabulary used in a predetermined set of target queries, is that we utilize a standardized external reference ontology, in which we extract a module related to the underlying data. Additionally, we don't assume our relational database to be denormalized and without integrity constraints.

In recent years, module extraction and modularization for ontologies have received significant attention in the literature, where different notions of modules and various algorithms to extract these modules have been proposed (Vescovo et al. 2020; Matentzoglu, Parsia, and Sattler 2018; Chen et al. 2019; Konev et al. 2013). In this work, we follow the syntactic locality-based approach of (Grau et al. 2009), but other notions, such as model-theoretic inseparability of (Konev et al. 2013), may also be considered.

Several different notions of acyclicity as sufficient conditions for chase termination have been studied in the literature, e.g., weak acyclicity (Fagin et al. 2005), super-weak acyclicity (Marnette 2009), stratification (Deutsch, Nash, and Remmel 2008), and C-stratification (Meier, Schmidt, and Lausen 2009). Here, in Theorem 1, we showed that every acyclic $\mathcal{ELH}^{fdr}$ terminology gives rise to a C-stratified set of tgds and egds, while those sets of tgds and egds need not be super-weakly acyclic. (Grau et al. 2013) introduced the notions of *model-faithful acyclicity* (MFA) and *model-summarising acyclicity* (MSA), showing that these notions guarantee the termination of the *Skolem chase*, and also that they properly contain the notion of super-weak acyclicity. However, (Grau et al. 2013) did not offer a comparison between MFA/MSA and C-stratification. Note that termination of the Skolem chase implies the termination of the standard chase, but not the other way around. It remains to be investigated whether the chase termination for acyclic $\mathcal{ELH}^{fdr}$ can also be derived from the results about MFA or MSA.

(Konev et al. 2012) consider acyclic $\mathcal{ELH}^{dr}$ terminologies following the classical acyclicity definition for terminologies (Baader et al. 2017). The main difference with the notion proposed here is that they do not pose any constraints on domain/range restrictions, which we have noted can destroy chase termination. Finally, (Mei et al. 2009) study query answering over acyclic ontologies of the $\mathcal{EL}$ family, but without range restrictions and functionality assertions. Unlike our chase-based approach to query answering, they only partially materialize the data and perform query rewriting to filter unsound answers. This is known as the combined approach (Kontchakov et al. 2011) and is mainly proposed for ontologies that may admit an infinite chase.

Concluding Remarks

We have presented an open-source framework that combines methods from different AI communities for ontology-enriched query answering. Along the way, we have identified a new class of acyclic $\mathcal{ELH}$ ontologies with domain/range restrictions and restricted functional roles, which we have proven to be C-stratified, implying chase termination in polynomial time. Our experimental evaluation demonstrates that our framework can discover additional answers and support new queries over a given medical database, while increasing the database size by only 24% (15 MB). Presently, we are exploring applications of our framework to domains other than medical, such as finance.

References

Baader, F.; Horrocks, I.; Lutz, C.; and Sattler, U. 2017. *An Introduction to Description Logic*. Cambridge University Press.

Benedikt, M.; Konstantinidis, G.; Mecca, G.; Motik, B.; Papotti, P.; Santoro, D.; and Tsamoura, E. 2017. Benchmarking the Chase. In *PODS*, 37–52.

Bienvenu, M. 2016. Ontology-Mediated Query Answering: Harnessing Knowledge to Get More from Data. In *IJCAI*, 4058–4061.

Calvanese, D.; Cogrel, B.; Komla-Ebri, S.; Kontchakov, R.; Lanti, D.; Rezk, M.; Rodriguez-Muro, M.; and Xiao, G. 2017. Ontop: Answering SPARQL queries over relational databases. *Semantic Web* 8(3): 471–487.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rodriguez-Muro, M.; Rosati, R.; Ruzzi, M.; and Savo, D. F. 2011. The MASTRO system for ontology-based data access. *Semantic Web* 2(1): 43–53.

Chen, J.; Alghamdi, G.; Schmidt, R. A.; Walther, D.; and Gao, Y. 2019. Ontology Extraction for Large Ontologies via Modularity and Forgetting. In *K-CAP*, 45–52.

de Uña, D.; Rümmele, N.; Gange, G.; Schachte, P.; and Stuckey, P. J. 2018. Machine Learning and Constraint Programming for Relational-To-Ontology Schema Mapping. In *IJCAI*, 1277–1283.

Deutsch, A.; Nash, A.; and Remmel, J. B. 2008. The chase revisited. In *PODS*, 149–158.

Efthymiou, V.; Hassanzadeh, O.; Rodriguez-Muro, M.; and Christophides, V. 2017. Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings. In *ISWC*, 260–277.

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

Giacomo, G. D.; Lembo, D.; Lenzerini, M.; Poggi, A.; and Rosati, R. 2018. Using Ontologies for Semantic Data Integration. In *A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*, volume 31 of *Studies in Big Data*, 187–202. Springer International Publishing.

Grau, B. C.; Horrocks, I.; Kazakov, Y.; and Sattler, U. 2008. Modular Reuse of Ontologies: Theory and Practice. *J. Artif. Intell. Res.* 31: 273–318.

Grau, B. C.; Horrocks, I.; Kazakov, Y.; and Sattler, U. 2009. Extracting Modules from Ontologies: A Logic-Based Approach. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*, 159–186. Springer.

Grau, B. C.; Horrocks, I.; Krötzsch, M.; Kupke, C.; Magka, D.; Motik, B.; and Wang, Z. 2013. Acyclicity Notions for Existential Rules and Their Application to Query Answering in Ontologies. *J. Artif. Intell. Res.* 47: 741–808.

Hernich, A.; Lutz, C.; Papacchini, F.; and Wolter, F. 2018. Horn-Rewritability vs PTime Query Evaluation in Ontology-Mediated Querying. In *IJCAI*, 1861–1867.

Hovland, D.; Kontchakov, R.; Skjæveland, M. G.; Waaler, A.; and Zakharyaschev, M. 2017. Ontology-Based Data Access to Slegge. In *ISWC*, 120–129.

Jiménez-Ruiz, E.; and Grau, B. C. 2011. LogMap: Logic-Based and Scalable Ontology Matching. In *ISWC*, 273–288.

Kharlamov, E.; Jiménez-Ruiz, E.; Pinkel, C.; Rezk, M.; Skjæveland, M. G.; Soylu, A.; Xiao, G.; Zheleznyakov, D.; Giese, M.; Horrocks, I.; and Waaler, A. 2015. Optique: Ontology-Based Data Access Platform. In *ISWC Posters & Demonstrations Track*, volume 1486 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Knoblock, C. A.; Szekely, P. A.; Ambite, J. L.; Goel, A.; Gupta, S.; Lerman, K.; Muslea, M.; Taheriyan, M.; and Mallick, P. 2012. Semi-automatically Mapping Structured Sources into the Semantic Web. In *ESWC*, 375–390.

Konev, B.; Ludwig, M.; Walther, D.; and Wolter, F. 2012. The Logical Difference for the Lightweight Description Logic EL. *J. Artif. Intell. Res.* 44: 633–708.

Konev, B.; Lutz, C.; Walther, D.; and Wolter, F. 2013. Model-theoretic inseparability and modularity of description logic ontologies. *Artif. Intell.* 203: 66–103.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2011. The Combined Approach to Ontology-Based Data Access. In *IJCAI*, 2656–2661.

Lei, C.; Özcan, F.; Quamar, A.; Mittal, A. R.; Sen, J.; Saha, D.; and Sankaranarayanan, K. 2018. Ontology-Based Natural Language Query Interfaces for Data Exploration. *IEEE Data Eng. Bull.* 41(3): 52–63.

Marnette, B. 2009. Generalized schema-mappings: from termination to tractability. In *PODS*, 13–22.

Matentzoglu, N.; Parsia, B.; and Sattler, U. 2018. OWL Reasoning: Subsumption Test Hardness and Modularity. *J. Autom. Reason.* 60(4): 385–419.

Mei, J.; Liu, S.; Xie, G. T.; Kalyanpur, A.; Fokoue, A.; Ni, Y.; Li, H.; and Pan, Y. 2009. A Practical Approach for Scalable Conjunctive Query Answering on Acyclic EL^+ Knowledge Base. In *ISWC*, 408–423.

Meier, M.; Schmidt, M.; and Lausen, G. 2009. On Chase Termination Beyond Stratification. *PVLDB* 2(1): 970–981.

Nenov, Y.; Piro, R.; Motik, B.; Horrocks, I.; Wu, Z.; and Banerjee, J. 2015. RDFox: A Highly-Scalable RDF Store. In *ISWC*, 3–20.

Quamar, A.; Lei, C.; Miller, D.; Özcan, F.; Kreulen, J.; Moore1, R. J.; and Efthymiou, V. 2020. An Ontology-Based Conversation System for Knowledge Bases. In *SIGMOD*, 361–376.

Rodriguez-Muro, M.; Kontchakov, R.; and Zakharyaschev, M. 2013. Ontology-Based Data Access: Ontop of Databases. In *ISWC*, 558–573.

Vescovo, C. D.; Horridge, M.; Parsia, B.; Sattler, U.; Schneider, T.; and Zhao, H. 2020. Modular Structures and Atomic Decomposition in Ontologies. *J. Artif. Intell. Res.* 69: 963–1021.

Xiao, G.; Calvanese, D.; Kontchakov, R.; Lembo, D.; Poggi, A.; Rosati, R.; and Zakharyaschev, M. 2018. Ontology-Based Data Access: A Survey. In *IJCAI*, 5511–5519.