

Guarded Ontology-Mediated Queries Distributing Over Components*

Pablo Barceló¹, Gerald Berger², and Andreas Pieris³

¹ Center for Semantic Web Research & DCC, University of Chile
pbarcelo@dcc.uchile.cl

² Institute of Information Systems, TU Wien gberger@dbai.tuwien.ac.at

³ School of Informatics, University of Edinburgh apieris@inf.ed.ac.uk

1 Introduction

Ontology-based data access (OBDA) has recently emerged as a promising application of knowledge representation and reasoning technologies in information management systems. The aim of OBDA is to facilitate access to data that is significantly heterogeneous and incomplete. This is achieved via an ontology that provides a unified conceptual view of the data, and makes it accessible via database queries formulated solely in the vocabulary of the ontology. The actual database query and the ontology can be seen as two components of one composite query, called *ontology-mediated query* [8]. OBDA can then be realized as the problem of answering ontology-mediated queries.

An important topic for declarative database query languages, and in particular (extensions of) Datalog, is coordination-free evaluation. This has its roots in *declarative networking* [15], an approach where distributed computations are programmed using Datalog-based formalisms. In this setting, programs (or queries) are specified over a global schema, and are executed by multiple computing nodes over which the database is distributed. These nodes can perform local computations, and also communicate asynchronously with each other via messages. The model assumes that messages can never be lost but can be arbitrarily delayed. An intrinsic source of inefficiency in such systems are the global barriers raised by the need for synchronization in computing the result of queries. This has motivated a fruitful line of research for isolating classes of queries that can be evaluated in a coordination-free manner [1–4, 16].

It is natural to ask whether the results on coordination-free evaluation for declarative database query languages can be transferred to ontology-mediated queries. Undoubtedly, a positive answer to this question, apart from possible applications to declarative networking, will significantly contribute towards more efficient procedures for OBDA, since it will enable distributed ontology-mediated query evaluation in a coordination-free way. The goal of the present work is to initiate the study of coordination-free evaluation for ontology-mediated queries, and give strong indications that the answer to the above challenging question is affirmative.

Distribution Over Components. More concretely, we focus on the question whether the answer to an ontology-mediated query can be computed by parallelizing it over the

* This short paper is based on [6, 7].

(maximally connected) components of the database, i.e., whether the query *distributes over components*. In other words, given an ontology-mediated query Q , the question is whether for every database D the answer to Q over D , denoted $Q(D)$, coincides with $\bigcup_{1 \leq i \leq n} Q(D_i)$, where D_1, \dots, D_n are the components of D . In this case, $Q(D)$ can be computed without any communication over a network using a distribution where every computing node is assigned some of the connected components of the database, and every component is assigned to at least one computing node. The notion of distribution over components has been introduced in [2], and explicitly considered for Datalog queries in [3]. It has been shown that *connected* Datalog, that is, the fragment of Datalog where all rule-bodies are connected, provides an effective syntax for Datalog queries that distribute over components, while the problem of deciding whether a Datalog query distributes over components is undecidable.

Aims and Objectives. As said, this work concentrates on ontology-mediated queries and their distribution over components. We focus on *guarded* ontology-mediated queries where the database query is a conjunctive query and the ontology is formulated using guarded existential rules (a.k.a. tuple-generating dependencies and Datalog[±] rules), that is, sentences of the form $\forall \bar{x} \forall \bar{y} (\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z}))$ where φ, ψ are conjunctions of atoms with φ being guarded, i.e., it has an atom, called *guard*, that contains all the variables $(\bar{x} \cup \bar{y})$ [10, 11]. It is widely accepted that the class of guarded existential rules forms a natural and convenient way for modeling ontologies, which generalizes prominent description logics such as \mathcal{ELHI} [5]. The goal of this work is to answer the following questions: (1) Can we characterize the fragment of guarded ontology-mediated queries that distribute over components via the notion of connectedness, i.e., when the rule-bodies and the conjunctive query are connected? (2) What is the complexity of deciding whether a guarded ontology-mediated query distributes over components?

Our Results. Our results can be summarized as follows:

- The fragment of guarded ontology-mediated queries that distribute over components has the same expressive power as the fragment of guarded ontology-mediated queries where the conjunctive query is connected. Notice that the body of a guarded existential rule is always connected due to the existence of the guard atom. Thus, this result provides a positive answer to the first question above.
- Regarding the second question, we show that deciding whether a guarded ontology-mediated query distributes over components is strongly related to a classical static analysis task, namely *containment*. We show that the problem in question is feasible in polynomial time assuming access to an oracle that can solve containment for guarded ontology-mediated queries. We then show that the containment problem is feasible in 2EXPTIME, which immediately implies that distribution over components is also feasible in 2EXPTIME. A matching lower bound can be easily shown by exploiting the fact that evaluation of guarded OMQs is 2EXPTIME-hard [10].

2 Distribution of Queries and Connectedness

Let us first recall the basics on ontology-mediated querying. An *ontology-mediated query* (OMQ) over a (relational) schema \mathbf{S} is a triple (\mathbf{S}, Σ, q) , where Σ is a set of

existential rules (the ontology), and q is a conjunctive query (CQ). A *guarded* OMQ is an OMQ as the one above where Σ is a set of guarded existential rules; we write $(\mathbb{G}, \mathbb{CQ})$ for the OMQ language that consists of all guarded OMQs.⁴ The semantics of an OMQ is given in terms of *certain answers*. The *evaluation* of $Q = (\mathbf{S}, \Sigma, q)$ over an \mathbf{S} -database D , which is a finite set of atoms of the form $R(\bar{t})$, where $R \in \mathbf{S}$ and \bar{t} is a tuple of constants, denoted $Q(D)$, is defined as the certain answers to q w.r.t. D and Σ . Equivalently, $Q(D)$ is the evaluation of q over the *canonical instance* of D and Σ that can be constructed by the well-known chase procedure. Recall that the chase adds new atoms to D as dictated by Σ until the final result, written $\text{chase}(D, \Sigma)$, satisfies all the rules of Σ . For more details on ontology-mediated queries see, e.g., [7].

The notion of distribution over components has been introduced in [2], and it states that the answer to a query can be computed by parallelizing it over the (maximally connected) components of the input database. But let us first make precise what a component is. A set A of atoms is *connected* if for all terms c, d occurring in A there exists a sequence $\alpha_1, \dots, \alpha_n$ of atoms in A such that c occurs in α_1 , d occurs in α_n , and α_i, α_{i+1} have at least one term in common, for each $i \in \{1, \dots, n-1\}$. We call $B \subseteq A$ a *component* of A if (i) B is connected, and (ii) for every $\alpha \in A \setminus B$, $B \cup \{\alpha\}$ is not connected. Let $\text{co}(A)$ be the set of components of A . We are now ready to introduce the notion of distribution over components. Consider an OMQ $Q = (\mathbf{S}, \Sigma, q)$. We say that Q *distributes over components* if $Q(D) = Q(D_1) \cup \dots \cup Q(D_n)$, where $\text{co}(D) = \{D_1, \dots, D_n\}$, for every \mathbf{S} -database D . Roughly, the centralized answer to Q w.r.t. D is precisely obtained when we parallelize Q over the components of D . Let $\mathbb{D}\text{IST}$ be the class of OMQs that distribute over components.

We proceed to characterize the expressive power of the fragment of $(\mathbb{G}, \mathbb{CQ})$ that distributes over components. This is done via connectedness of CQs. A CQ q is *connected* if the set of atoms occurring in q is connected; we write $\mathbb{C}\mathbb{C}\mathbb{Q}$ for the class of connected CQs. The main result of this section states that every guarded OMQ that distributes over components is equivalent to a connected guarded OMQ. Given two OMQ languages \mathbb{O}_1 and \mathbb{O}_2 , we write $\mathbb{O}_1 = \mathbb{O}_2$ to state that they are equally expressive, i.e., for every query $Q_1 \in \mathbb{O}_1$ over \mathbf{S} we can construct a query $Q_2 \in \mathbb{O}_2$ over \mathbf{S} such that $Q_1(D) = Q_2(D)$, for every \mathbf{S} -database D , and vice versa. Then:

Theorem 1. $(\mathbb{G}, \mathbb{CQ}) \cap \mathbb{D}\text{IST} = (\mathbb{G}, \mathbb{C}\mathbb{C}\mathbb{Q})$.

For the (\supseteq) direction we show that connectedness ensures distribution over components. This is a consequence of the fact that, given a query $Q = (\mathbf{S}, \Sigma, q) \in (\mathbb{G}, \mathbb{C}\mathbb{C}\mathbb{Q})$, for every \mathbf{S} -database D with $\text{co}(D) = \{D_1, \dots, D_m\}$, $\text{chase}(D, \Sigma)$ can be partitioned into $\{I_1, \dots, I_m\}$ such that, for each $i \in \{1, \dots, m\}$, I_i depends solely on D_i . Consider now a query $Q = (\mathbf{S}, \Sigma, q(\bar{x})) \in (\mathbb{G}, \mathbb{CQ}) \cap \mathbb{D}\text{IST}$. Observe that if Q is *unsatisfiable*, i.e., there is no \mathbf{S} -database D such that $Q(D) \neq \emptyset$, then the claim holds trivially; simply choose an arbitrary unsatisfiable query from $(\mathbb{G}, \mathbb{C}\mathbb{C}\mathbb{Q})$. The interesting case is when Q is *satisfiable*. Assuming that $\{q_1, \dots, q_k\}$ are the components of q , we can show that there exists $i \in \{1, \dots, k\}$ such that $Q_i = (\mathbf{S}, \Sigma, q_i(\bar{x}))$ is well-defined and equivalent to Q . Since $Q_i \in (\mathbb{G}, \mathbb{C}\mathbb{C}\mathbb{Q})$ the claim follows.

⁴ Notice that, in general, OMQs are defined for arbitrary ontology languages based on first-order logic (not only on existential rules), and arbitrary query languages (not only CQs).

3 Deciding Distribution Over Components

In this section we focus on the problem of deciding whether a guarded OMQ distributes over components. In fact, this problem can be defined for an OMQ language based on an arbitrary class \mathbb{C} of existential rules:

PROBLEM : $\text{Dist}(\mathbb{C}, \mathbb{CQ})$
INPUT : An OMQ $Q \in (\mathbb{C}, \mathbb{CQ})$.
QUESTION : Does Q distribute over components?

Our goal is to pinpoint the complexity of $\text{Dist}(\mathbb{G}, \mathbb{CQ})$. Towards this direction, we first show that it is closely related to the crucial task of containment: given two OMQs $Q_1, Q_2 \in (\mathbb{G}, \mathbb{CQ})$ over a schema \mathbf{S} , decide whether $Q_1(D) \subseteq Q_2(D)$, for every \mathbf{S} -database D , written as $Q_1 \subseteq Q_2$. By exploiting Theorem 1 we can show the following:

Proposition 1. *Let $Q = (\mathbf{S}, \Sigma, q(\bar{x})) \in (\mathbb{G}, \mathbb{CQ})$. The following are equivalent:*

1. Q distributes over components.
2. Q is unsatisfiable or there exists $\hat{q}(\bar{x}) \in \text{co}(q)^5$ such that $(\mathbf{S}, \Sigma, \hat{q}(\bar{x})) \subseteq Q$.

Notice that checking unsatisfiability can be easily reduced to containment. Therefore, the above results essentially states that $\text{Dist}(\mathbb{G}, \mathbb{CQ})$ is feasible in polynomial time assuming access to a \mathcal{C} oracle, where \mathcal{C} is a complexity class powerful enough for checking containment among guarded OMQs. The latter is a highly non-trivial problem, and we have recently shown that is 2EXPTIME-complete [6]. To obtain the 2EXPTIME upper bound, we make use of techniques based on *two-way alternating parity automata on trees* (2WAPA) [12]. We first show that if Q_1 and Q_2 are guarded OMQs such that Q_1 is not contained in Q_2 , then this is witnessed over a class of “tree-like” databases that can be represented as the set of trees accepted by a 2WAPA \mathfrak{A} . We then build a 2WAPA \mathfrak{B} with exponentially many states that recognizes those trees accepted by \mathfrak{A} that represent witnesses to non-containment of Q_1 in Q_2 . Hence, Q_1 is contained in Q_2 iff \mathfrak{B} accepts no tree. Since the emptiness problem for 2WAPA is feasible in exponential time in the number of states [12], we obtain that containment for guarded OMQs is in 2EXPTIME. A matching lower bound follows from [9]. Thus, Proposition 1 together with the fact that containment for guarded OMQs is in 2EXPTIME implies that $\text{Dist}(\mathbb{G}, \mathbb{CQ})$ is in 2EXPTIME, while a matching lower bound can be shown by exploiting the fact that evaluation of $(\mathbb{G}, \mathbb{CQ})$ queries is 2EXPTIME-hard [10]. Then:

Theorem 2. $\text{Dist}(\mathbb{G}, \mathbb{CQ})$ is 2EXPTIME-complete.

4 Next Steps

Here is a couple of interesting open problems that we are planning to tackle: (i) distribution over components in the presence of equality and denial constraints is not well

⁵ This denotes the set of components of the set of atoms in q , or simply, the components of q .

understood, and (ii) we do not know how distribution over components behaves in the case of guarded OMQs enriched with non-monotonic negation [13, 14].

Acknowledgements. Barceló is supported by the Millenium Nucleus Center for Semantic Web Research under grant NC120004. Berger is funded by the Austrian Science Fund (FWF), project W1255-N23, and a DOC Fellowship of the Austrian Academy of Sciences. Pieris is supported by the EPSRC Programme Grant EP/M025268/ “VADA: Value Added Data Systems – Principles and Architecture”.

References

1. Alvaro, P., Conway, N., Hellerstein, J.M., Maier, D.: Blazes: Coordination analysis for distributed programs. In: ICDE. pp. 52–63 (2014)
2. Ameloot, T.J., Ketsman, B., Neven, F., Zinn, D.: Weaker forms of monotonicity for declarative networking: A more fine-grained answer to the calm-conjecture. In: PODS. pp. 64–75 (2014)
3. Ameloot, T.J., Ketsman, B., Neven, F., Zinn, D.: Datalog queries distributing over components. In: ICDT. pp. 308–323 (2015)
4. Ameloot, T.J., Neven, F., den Bussche, J.V.: Relational transducers for declarative networking. *J. ACM* 60(2), 15 (2013)
5. Baader, F., Brandt, S., Lutz, C.: Pushing the el envelope. In: Proc. of IJCAI. pp. 364–369 (2005)
6. Barceló, P., Berger, G., Pieris, A.: Containment for rule-based ontology-mediated queries. CoRR abs/1703.07994 (2017), <http://arxiv.org/abs/1703.07994>
7. Berger, G., Pieris, A.: Ontology-mediated queries distributing over components. In: IJCAI. pp. 943–949 (2016)
8. Bienvenu, M., ten Cate, B., Lutz, C., Wolter, F.: Ontology-based data access: A study through disjunctive datalog, csp, and MMSNP. *ACM Trans. Database Syst.* 39(4), 33:1–33:44 (2014)
9. Bienvenu, M., Hansen, P., Lutz, C., Wolter, F.: First order-rewritability and containment of conjunctive queries in horn description logics. In: IJCAI. pp. 965–971 (2016)
10. Cali, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res.* 48, 115–174 (2013)
11. Cali, A., Gottlob, G., Lukasiewicz, T.: A general Datalog-based framework for tractable query answering over ontologies. *J. Web Sem.* 14, 57–83 (2012)
12. Cosmadakis, S.S., Gaifman, H., Kanellakis, P.C., Vardi, M.Y.: Decidable optimization problems for database logic programs (preliminary report). In: STOC. pp. 477–490 (1988)
13. Gottlob, G., Hernich, A., Kupke, C., Lukasiewicz, T.: Stable model semantics for guarded existential rules and description logics. In: KR (2014)
14. Hernich, A., Kupke, C., Lukasiewicz, T., Gottlob, G.: Well-founded semantics for extended Datalog and ontological reasoning. In: PODS. pp. 225–236 (2013)
15. Loo, B.T., Condie, T., Garofalakis, M.N., Gay, D.E., Hellerstein, J.M., Maniatis, P., Ramakrishnan, R., Roscoe, T., Stoica, I.: Declarative networking. *Commun. ACM* 52(11), 87–95 (2009)
16. Zinn, D., Green, T.J., Ludäscher, B.: Win-move is coordination-free (sometimes). In: ICDT. pp. 99–113 (2012)