HAL open science

Parallelisable Existential Rules: a Story of Pieces

Maxime Buron, Marie-Laure Mugnier, Michaël Thomazo

To cite this version:

Maxime Buron, Marie-Laure Mugnier, Michaël Thomazo. Parallelisable Existential Rules: a Story of Pieces. KR 2021 - 18th International Conference on Principles of Knowledge Representation and Reasoning, Nov 2021, Virtual, Vietnam. pp.163-173, 10.24963/kr.2021/16 . hal-03405745

HAL Id: hal-03405745

https://inria.hal.science/hal-03405745v1

Submitted on 27 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Parallelisable Existential Rules: a Story of Pieces

Maxime Buron[1] , **Marie-Laure Mugnier**[2] , **Michaël Thomazo**[3]
[1] University of Oxford, United Kingdom
[2] LIRMM, Inria, University of Montpellier, CNRS, France
[3] Inria, DI ENS, ENS, CNRS, PSL University & Inria, France
maxime.buron@cs.ox.ac.uk, michael.thomazo@inria.fr, mugnier@lirmm.fr

Abstract

In this paper, we consider existential rules, an expressive formalism well suited to the representation of ontological knowledge and data-to-ontology mappings in the context of ontology-based data integration. The chase is a fundamental tool to do reasoning with existential rules as it computes all the facts entailed by the rules from a database instance. We introduce parallelisable sets of existential rules, for which the chase can be computed in a single breadth-first step from any instance. The question we investigate is the characterization of such rule sets. We show that parallelisable rule sets are exactly those rule sets both bounded for the chase and belonging to a novel class of rules, called pieceful. The pieceful class includes in particular frontier-guarded existential rules and (plain) datalog. We also give another characterization of parallelisable rule sets in terms of rule composition based on rewriting.

1 Introduction

Ontology-based data access (OBDA) systems aim at facilitating data querying through a conceptual layer formalized by an ontology (Poggi et al. 2008; Xiao et al. 2018). They rely on a three-level architecture comprising the ontology, the data sources and the mapping between the two. The key idea is that a user expresses queries at a conceptual level, and the system translates these queries into queries on the data via the mapping, while integrating ontological reasoning.

When we abstract away from data sources and mappings, we obtain the fundamental *ontology-based query answering* problem, which takes as input an ontology $\mathcal{O}$, an instance (or set of facts) I and a (Boolean) conjunctive query q, both expressed in the vocabulary of $\mathcal{O}$, and asks whether $I, \mathcal{O} \models q$. Ontological knowledge is typically represented in *description logics* (*e.g.*, (Baader et al. 2017; Bienvenu and Ortiz 2015)) or *existential rules* (*e.g.*, (Calì, Gottlob, and Lukasiewicz 2009; Baget et al. 2009)) and we shall consider the latter language in this paper. Existential rules are an extension of first-order function-free Horn rules allowing for existentially quantified variables in the rule heads (*e.g.*, $\forall x(s(x) \rightarrow \exists z\, t(x,z))$), which makes them able to infer the existence of unknown individuals. They generalise datalog and most description logics used to do reasoning on data, namely Horn description logics.

Two dual techniques are used to solve the ontology-based query answering problem: the *chase*, which enriches I by performing a fixpoint computation with $\mathcal{O}$ until a canonical model of I and $\mathcal{O}$ is obtained (then q is asked on the result of the chase), and *query rewriting*, where q is rewritten with $\mathcal{O}$ into a query q', such that for all instance I, holds $I, \mathcal{O} \models q$ if and only if holds $I \models q'$. Query answering is undecidable with general existential rules, however there are expressive subclasses ensuring the termination of either technique.

In the OBDA paradigm, the instance I is not materialized, but virtually defined by the mapping and the data. Precisely, an OBDA specification is given by an ontology $\mathcal{O}$, a relational schema $\mathcal{S}$ and a mapping $\mathcal{M}$ from $\mathcal{S}$ to $\mathcal{O}$ (Lenzerini 2018). The mapping is itself composed of assertions of the form $q_S(\mathbf{x}) \rightarrow q_O(\mathbf{x})$, where q_S is a query on schema $\mathcal{S}$ and q_O is a conjunctive query on the vocabulary of $\mathcal{O}$, both with tuple of answer variables $\mathbf{x}$. When q_S is also a conjunctive query, or a relational view, a mapping assertion can be seen as an existential rule. Then, the virtual instance $I_{(D,\mathcal{M})}$, associated with a database D (on $\mathcal{S}$) and the mapping $\mathcal{M}$, is the set of facts that would be obtained by chasing D with $\mathcal{M}$. Note that only a single (breadth-first) step of the chase is required here, as bodies and heads of mapping assertions are on disjoint sets of predicates. Since $I_{(D,\mathcal{M})}$ is virtual, an incoming query has to be rewritten, first with $\mathcal{O}$, then with $\mathcal{M}$, which yields a query directly asked on D. As rewriting is performed at query time, speeding up this process is a crucial issue. In particular, query rewriting with $\mathcal{O}$ is a recursive process, which is not the case with $\mathcal{M}$. For very lightweight ontology languages (the DL-Lite family or the W3C language RDFS), a practically efficient approach consists of compiling (part of) the ontological reasoning into the mapping, so that the rewriting step with $\mathcal{O}$ can be avoided or drastically reduced (Kontchakov et al. 2014; Buron et al. 2020b). In these settings, each mapping assertion can be processed independently, the mapping head being enriched with knowledge it entails. Whether such technique can be extended to more expressive languages is an open issue, which motivated the work presented here.

Consider an OBDA setting where existential rules are used as a uniform language to express both the ontology and the mapping. Compiling ontological reasoning into the mapping can be seen as computing a new mapping $\mathcal{M}'$ such that query rewriting with $\mathcal{O}$ and $\mathcal{M}$ is reduced to query rewriting with $\mathcal{M}'$. From a dual viewpoint, for any database D, the instance $I_{(D,\mathcal{M}')}$ is equivalent to the chase of $I_{(D,\mathcal{M})}$ with

$\mathcal{O}$. The next example illustrates the approach.

Example 1. *Let $\mathcal{M} = \{M_1, M_2\}$ and $\mathcal{O} = \{R_1, R_2\}$, where s_i and t_i denote predicates from $\mathcal{S}$ and $\mathcal{O}$, respectively, and universal quantifiers are omitted:*
$M_1 = s_1(x, y) \rightarrow t_1(x, y)$
$M_2 = s_2(x) \rightarrow t_2(x)$
$R_1 = t_2(x) \rightarrow \exists z\, t_3(x, z)$
$R_2 = t_1(x, y) \wedge t_3(x, z) \rightarrow t_4(y)$
Here, the ontology can be compiled into the mapping, which yields $\mathcal{M}' = \mathcal{M} \cup \{M_3, M_4\}$, where:
$M_3 = s_2(x) \rightarrow \exists z\, t_3(x, z)$
$M_4 = s_1(x, y) \wedge s_2(x) \rightarrow t_4(y)$.
Intuitively, $\mathcal{M}'$ is obtained by composing the rules from $\mathcal{M} \cup \mathcal{O}$ until a fixpoint is reached (see Sect. 4), then keeping only mapping assertions, i.e., *rules whose all body predicates are in $\mathcal{S}$. We can check that, for any database D on $\mathcal{S}$, a single breadth-first step of the chase of D with $\mathcal{M}'$ suffices to produce the chase of D with $\mathcal{M} \cup \mathcal{O}$.*

With the aim of developing compilation techniques for OBDA systems based on existential rules, we asked ourselves the following question: under which conditions on the rules can the chase be simulated in a single step?

We formalize the desired property by the notion of *parallelisable* existential rule sets. Informally, a (finite) rule set $\mathcal{R}$ is parallelisable if there exists a finite rule set $\mathcal{R}'$ able to produce (an equivalent superset of) the chase of $\mathcal{R}$ in a *single* breadth-first step, independently from any instance. The question we investigate in this paper is how to characterize such rule sets. Our main results are the following.

- Clearly, *boundedness*, which expresses that the number of chase steps is bounded independently from any instance, is a necessary condition for parallelisability. This notion has long been studied for datalog (Hillebrand et al. 1995) and more recently for existential rules (Bourhis et al. 2019; Delivorias et al. 2021). While boundedness is equivalent to parallelisability in the case of datalog, it does not ensure the parallelisability of an existential rule set. This leads us to define a new class of existential rules, namely *pieceful*. We show that parallelisable rule sets are exactly those sets that are both bounded and pieceful.
- The novel pieceful class has an interest in itself, as it generalizes datalog as well as a main class of existential rules, namely frontier-guarded, which itself covers some prominent description logics (see the last section for details).
- Piecefulness is based on the behavior of rules during the chase. Adopting the viewpoint of query rewriting, we study an operator of rule composition (that we call *existential composition*) based on so-called piece-unifiers, a notion at the core of rewriting with existential rules. We show that any bounded and pieceful rule set can be parallelized by a finite set of composed rules.
- This allows us to provide another characterization of pieceful rule sets based on their behavior during rule composition. Roughly, a certain property (that we call *existential stability*) has to be fulfilled by every pair of rules of the set and preserved whenever composed rules are added to the set.
- Finally, noting that rule composition does not fit well with the behavior we intuitively expected, we introduce another rule composition operator (that we call *compact composition*). The result of this operator goes beyond the existential rule language. We show however that, when the stability property is satisfied, both kinds of compositions yield logically equivalent formulas.

We believe that these results open up many practical and theoretical perspectives, which we discuss in the last section. Full proofs that could not be included due to space restrictions are available in a technical report (Buron, Mugnier, and Thomazo 2021).

2 Preliminaries

Generalities. A vocabulary is a pair $\mathcal{V} = (\mathcal{P}, \mathcal{C})$, where $\mathcal{P}$ is a finite set of predicates and $\mathcal{C}$ is a possibly infinite set of constants. A *term* on $\mathcal{V}$ is a constant from $\mathcal{C}$ or a variable. An *atom* on $\mathcal{V}$ has the form $p(\mathbf{t})$ where $p \in \mathcal{P}$ is a predicate of arity n and $\mathbf{t}$ is a tuple of terms on $\mathcal{V}$ with $|\mathbf{t}| = n$. An atom is *ground* if it has no variable. Given an atom or set of atoms S, we denote by $\mathrm{var}(S)$, $\mathrm{const}(S)$ and $\mathrm{term}(S)$ its sets of variables, constants and terms, respectively. We will often see a tuple $\mathbf{x}$ of pairwise distinct variables as a set. We denote by $\models$ the classical logical consequence. Given two sets of atoms S_1 and S_2, a *homomorphism* h from S_1 to S_2 is a substitution of $\mathrm{var}(S_1)$ by $\mathrm{term}(S_2)$ such that $h(S_1) \subseteq S_2$ (we say that S_1 *maps* to S_2 by h).

Instances and Rules. An *instance* is a finite set of ground atoms. Any finite set of atoms S can be turned into an instance, denoted by $\mathrm{freeze}(S)$, by *freezing* its variables, *i.e.*, bijectively renaming each variable by a fresh constant. An *extended instance* is a (possibly infinite) set of atoms, in which variables are classically called *nulls*. The associated (possibly infinite) formula is the existential closure of the conjunction of the atoms. An *existential rule* R (or simply *rule* hereafter) is a closed formula of the form

$$\forall \mathbf{x} \forall \mathbf{y}\ [\ B(\mathbf{x}, \mathbf{y}) \rightarrow \exists \mathbf{z}\ H(\mathbf{x}, \mathbf{z})\ ]$$

where B and H are non-empty and finite conjunctions of atoms on variables, respectively called the *body* and the *head* of the rule, denoted by $\mathrm{body(R)}$ and $\mathrm{head(R)}$, and $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$ are pairwise disjoint. We make the common assumption that rules do not contain constants, which simplifies technical tools. The set $\mathbf{x}$ is called the *frontier* of R and is denoted by $\mathrm{fr}(R)$. Its elements are called frontier variables. The set $\mathbf{z}$ is called the set of *existential variables* (of R) and is denoted by $\mathrm{exist}(R)$. An existential rule R is *datalog* if $|\mathrm{head(R)}| = 1$ and $\mathrm{exist}(R) = \emptyset$. For brevity, we often denote by $B \rightarrow H$ a rule with body B and head H. In the following, we denote by $\mathcal{R}$ a finite set of existential rules and assume w.l.o.g. that distinct rules in $\mathcal{R}$ have disjoint sets of variables. In our examples, we reuse variables for simplicity.

A rule $R = B \rightarrow H$ is *applicable* to a set of atoms S if there is a homomorphism π from B to S. The pair (R, π) is called a *trigger* for S. The application of R according to π (or: of the trigger (R, π)) produces a set of atoms obtained from $\mathrm{head(R)}$ by replacing each frontier variable x

with $\pi(x)$ and each existential variable with a fresh variable, called a *null*. We denote by π^{safe} this extension of π that "safely" renames existential variables; hence, atoms produced by distinct applications of the same rule have disjoint sets of nulls. The set of atoms resulting from the application of (R, π) to S is $\alpha(S, R, \pi) = S \cup \pi^{\text{safe}}(H)$.

An $\mathcal{R}$*-derivation* (from I to I_k) is a finite sequence $(I_0 = I), (R_1, \pi_1, I_1), \ldots, (R_k, \pi_k, I_k)$ such that for all $0 < i \leq k$, $R_i \in \mathcal{R}$, (R_i, π_i) is a trigger for I_{i-1} and $I_i = \alpha(I_{i-1}, R_i, \pi_i)$. When only the successive (extended) instances are needed, we note $(I_0 = I), I_1, \ldots, I_k$.

Chase. The chase builds a derivation from an instance by repeatedly applying rules until a fixpoint is reached. We rely on the *semi-oblivious* chase variant (in short, *so-chase*), in which two triggers (R, π_1) and (R, π_2) that coincide on the frontier of R produce exactly the same result (Marnette 2009). More precisely, we assume that nulls are named as follows: given a trigger (R, π), for all $z \in \text{exist}(R)$, $\pi^{\text{safe}}(z) = z_{(R, \pi_{|\text{fr}(R)})}$, where $\pi_{|\text{fr}(R)}$ denotes the restriction of π to the domain $\text{fr}(R)$. Hence, the name of a null created by a trigger (R, π) is based on R and $\pi_{|\text{fr}(R)}$, not π itself. We consider a breadth-first so-chase defined as follows: $\text{chase}_0(I, \mathcal{R}) = I$ and, for any $i > 0$,

$$\text{chase}_i(I, \mathcal{R}) = \text{chase}_{i-1}(I, \mathcal{R}) \cup \bigcup_{(R,\pi)} (\pi^{\text{safe}}(\text{head(R)}))$$

where (R, π) is any trigger for $\text{chase}_{i-1}(I, \mathcal{R})$ and for any $z \in \text{exist}(R)$, $\pi^{\text{safe}}(z) = z_{(R, \pi_{|\text{fr}(R)})}$. Finally, $\text{chase}_\infty(I, \mathcal{R}) = \bigcup_{i \geq 0} \text{chase}_i(I, \mathcal{R})$.

Example 2. *Let* $I = \{p(a, b)\}$ *and* $\mathcal{R} = \{p(x, y) \rightarrow \exists z\, p(x, z) \wedge A(z)\}$. *By the trigger* $t_1 = (R, \{x \mapsto a, y \mapsto b\})$, *we obtain* $\text{chase}_1(I, \mathcal{R}) = \{p(a, b), p(a, \nu), A(\nu)\}$ *with* $\nu = z_{(R, \{x \mapsto a\})}$. *The trigger* $t_2 = (R, \{x \mapsto a, y \mapsto \nu\})$ *then produces the same atoms as* t_1. *Finally,* $\text{chase}_\infty(I, \mathcal{R}) = \text{chase}_1(I, \mathcal{R})$.

Query Answering. A *conjunctive query* (CQ) is of the form $q(\mathbf{x}) = \exists \mathbf{y}\, \varphi(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ and $\mathbf{y}$ are disjoint tuples of variables, φ is a conjunction of atoms, and $\mathbf{x} \cup \mathbf{y} = var(\varphi)$; the free variables in φ (*i.e.*, $\mathbf{x}$) are called answer variables. A *Boolean CQ* has no free variables. A *union of conjunctive queries* (UCQ) is a disjunction of CQs that have the same arity $|\mathbf{x}|$. An (extended) instance I answers positively to a Boolean CQ q iff $I \models q$. More generally, a tuple of constants $\mathbf{c}$ is an *answer* to a CQ $q(\mathbf{x})$, with $|\mathbf{x}| = |\mathbf{c}|$, on I if $I \models q[\mathbf{c}]$, where $q[\mathbf{c}]$ is obtained by substituting the i-th variable in $\mathbf{x}$ by the i-th constant in $\mathbf{c}$. The *CQ answering* problem takes as input an instance I, a rule set $\mathcal{R}$, a query $q(\mathbf{x})$ and a tuple of constants $(\mathbf{c})$, with $|\mathbf{x}| = |\mathbf{c}|$, and asks whether $I, \mathcal{R} \models q[\mathbf{c}]$. It holds that $I, \mathcal{R} \models q[\mathbf{c}]$ iff there is an $\mathcal{R}$-derivation from I to I' such that $I' \models q[\mathbf{c}]$. Equivalently, $I, \mathcal{R} \models q[\mathbf{c}]$ iff there is k such that $\text{chase}_k(I, \mathcal{R}) \models q[\mathbf{c}]$.

Pieces. The notion of piece is key in this paper. Given a set of atoms S, a *piece* of S with respect to a set of terms T is a non-empty set $S' \subseteq S$ such that (1) for any atom $a \in S$, if a shares a term from T with some $a' \in S'$ then $a \in S'$, and (2) there is no strict subset of S' that satisfies (1), *i.e.*, S' is minimal. Intuitively, atoms of S are glued together by the terms of T, which yields pieces. Note that, for any set of atoms S and set of terms T, S can be partitioned into pieces w.r.t. T. Here, pieces are defined with respect to existential variables or to nulls, depending on the considered objects. By default, a piece of S is w.r.t. $\text{exist}(S)$ if S is a rule head and w.r.t. the nulls in S if S is an (extended) instance.

A rule is called *single-piece* if its head forms a single piece. Any rule can be decomposed into a (trivially) equivalent set of single-piece rules. For instance, a rule $r(x, y) \rightarrow \exists z_1 \exists z_2\, p(x, z_1) \wedge A(z_1) \wedge A(z_2) \wedge p(x, y)$ has a head with three pieces: $\{p(x, z_1), A(z_1)\}$, $\{A(z_2)\}$ and $\{p(x, y)\}$, hence can be decomposed into three single-piece rules: $r(x, y) \rightarrow \exists z_1\, p(x, z_1) \wedge A(z_1)$; $r(x, y) \rightarrow \exists z_2\, A(z_2)$ and $r(x, y) \rightarrow p(x, y)$. In the following, we assume that rules are single-piece. This assumption simplifies our setting, although all notions and results of this paper could be reformulated without making it.

Piece-Unifiers. Piece-unifiers are a generalization of classical unifiers that take care of existential variables in rule heads by unifying sets of atoms instead of single atoms (Baget et al. 2009). Query rewriting as well as rule composition (see Sect. 4) are based on this notion. In the definition below, we give a simplified version of piece-unifiers, which does not take constants into account. See, *e.g.*, (König et al. 2015) for details about piece-unifiers.

Given sets of atoms S and $S' \subseteq S$, the set of *separating variables* in S' w.r.t. S, denoted by $\text{sep}_S(S')$, is the set of variables that belong to both S' and $S \backslash S'$. Note that when S' is a piece of an extended instance I, $\text{sep}_I(S')$ is necessarily empty (as S' shares only constants with the rest of I).

Definition 1 (Piece-unifier)**.** *Let* S *be a set of atoms and* $R = B \rightarrow H$ *be a rule (both without constants) such that* $\text{var}(S) \cap \text{var}(B \cup H) = \emptyset$. *A* piece-unifier *of* S *with* R *(or with* H*) is a triple* $\mu = (S', H', u)$ *with* $S' \neq \emptyset$, $S' \subseteq S$, $H' \subseteq H$, *and* u *is a substitution of* $\text{fr}(R) \cup \text{var}(S')$ *by* $\text{var}(\text{head(R)})$ *such that:*

1. *For all* $x \in \text{fr}(R)$, $u(x) \in \text{fr}(R)$
2. *For all* $x \in \text{sep}_S(S')$, $u(x) \in \text{fr}(R)$
3. $u(S') = u(H')$.

Let S be a set of atoms and $\mu = (S', H', u)$ be a piece-unifier of S with $R : B \rightarrow H$. The *(direct) rewriting* of S w.r.t. μ is $\beta(S, R, \mu) = u(B) \cup u(S \setminus S')$. An $\mathcal{R}$*-rewriting* S_k of S is obtained by a finite sequence $(S_0 = S), \ldots, S_k$ such that, for all $0 < i \leq k$, S_i is a direct rewriting of S_{i-1} w.r.t. a piece-unifier of S_{i-1} with a (copy of a) rule from $\mathcal{R}$.

Example 3 (Piece-Unifier)**.** *Let* $R = A(x) \rightarrow \exists z\, p(x, z)$ *and* $S_1 = \{p(w, v), B(v)\}$. *There is no piece-unifier of* S_1 *with* R *since* v *is a separating variable of* $S_1' = \{p(w, v)\}$, *hence cannot be unified with* z. *Let* $S_2 = \{p(w_1, v), B(w_1), p(w_2, v), C(w_2)\}$. *The triple* $\mu = (S_2', H', u)$ *with* $S_2' = \{p(w_1, v), p(w_2, v)\}$, $H' = \{p(x, z)\}$ *and* $u = \{w_1, w_2 \mapsto x,\ v \mapsto z\}$ *is a piece-unifier of* S_2 *with* R, *which yields the direct rewriting* $\{A(x), B(x), C(x)\}$.

A fundamental property of piece-unifiers is the following: given any instance I, rule set $\mathcal{R}$ and Boolean CQ q, there

is an $\mathcal{R}$-derivation from I to an I_i such that q maps to I_i ($I_i \models q$) iff there is an $\mathcal{R}$-rewriting q' of q such that q' maps to I ($I \models q'$). Note that this property does not rely on rules being single-piece.

Fundamental Properties of Rule Sets. A rule set $\mathcal{R}$ is *so-chase finite* if for any instance I, there is k such that $\text{chase}_k(I, \mathcal{R}) = \text{chase}_\infty(I, \mathcal{R})$. A rule set $\mathcal{R}$ is a *finite unification set* (fus) if for any Boolean CQ q, there is a finite set Q of $\mathcal{R}$-rewritings of q, such that, for any $\mathcal{R}$-rewriting q' of q, there is $q'' \in Q$ that maps to q' (Baget et al. 2009). It is not hard to see that fus is equivalent to first-order rewritability (introduced in (Calvanese et al. 2007)): $\mathcal{R}$ is first-order rewritable if for any Boolean CQ q, there is a UCQ Q, such that for any instance I, it holds that $I, \mathcal{R} \models q$ iff $I \models Q$. Moreover, as several times remarked, fus is equivalent to the *bounded derivation depth property* (Calì, Gottlob, and Lukasiewicz 2009): for any Boolean CQ q, there is k such that for any instance I, if $I, \mathcal{R} \models q$ then $\text{chase}_k(I, \mathcal{R}) \models q$. Finally, a rule set is *so-bounded* if there is k (a bound) such that for any instance I, $\text{chase}_k(I, \mathcal{R}) = \text{chase}_\infty(I, \mathcal{R})$ (Delivorias et al. 2021). Clearly, when a rule set is so-bounded, it is also so-chase finite and fus. The reciprocal holds true for single-piece rules (follows from (Bourhis et al. 2019)). In the remaining, we will simply write chase and bounded in place of so-chase and so-bounded.

Example 4.
$\mathcal{R}_1 = \{A(x) \to \exists z\, p(x, z) \wedge A(z)\}$ is fus and not chase-finite.
$\mathcal{R}_2 = \{R_1 : p(x, y) \wedge p(y, z) \to p(x, z)\}$ is datalog (hence chase-finite) and not fus (hence not bounded).
$\mathcal{R}_3 = \mathcal{R}_2 \cup \{R_2 : p(x, y) \wedge p(u, z) \to p(x, z)\}$ is bounded. Note that the body of R_2 contains 'disconnected' atoms and all the atoms produced by R_1 are also produced by R_2; moreover, for any instance I, all the atoms producible by R_2 are produced at the first breadth-first step of the chase. That is why $\mathcal{R}_3$ is bounded with bound 1.

3 Characterizing Parallelisable Rule Sets

In this section, we define parallelisable rule sets and motivate the introduction of a new class of existential rules, namely pieceful. We show that parallelisable rule sets are exactly those rule sets that are both bounded and pieceful.

3.1 Parallelisable Rule Sets

Parallelisability intuitively means that a sound superset of the chase can be obtained in a single breadth-first step by a finite set of rules independent from any instance.

Definition 2 (Parallelisability)**.** *A set of rules $\mathcal{R}$ is* parallelisable *if there exists a finite rule set $\mathcal{R}'$ such that for any instance I:*

1. *there is an injective homomorphism from $\text{chase}_\infty(I, \mathcal{R})$ to $\text{chase}_1(I, \mathcal{R}')$;*
2. *there is a homomorphism from $\text{chase}_1(I, \mathcal{R}')$ to $\text{chase}_\infty(I, \mathcal{R})$.*

Such rule set $\mathcal{R}'$ is said to parallelise *$\mathcal{R}$.*

Note on this definition. A more powerful notion of parallelisability could be obtained by dropping the injectivity requirement in Point 1. Then, $\text{chase}_1(I, \mathcal{R}')$ would be equivalent to $\text{chase}_\infty(I, \mathcal{R})$ but would not necessarily include it. This notion could also be obtained without changing the definition but considering a stronger chase variant for $\text{chase}_\infty(I, \mathcal{R})$, namely the *core* chase, which produces a minimal canonical model of $I \cup \mathcal{R}$ (Deutsch, Nash, and Remmel 2008). However, the core chase is not monotonic and its result may even not be obtainable by any $\mathcal{R}$-derivation. We have chosen here to consider the well-behaved so-chase.

Proposition 1. *If $\mathcal{R}$ is parallelisable, then it is bounded.*

Proof. Let us assume that $\mathcal{R}$ is parallelised by $\mathcal{R}'$. For any instance I, $\text{chase}_1(I, \mathcal{R}')$ is finite because $\mathcal{R}'$ is finite. As there is an injection from $\text{chase}_\infty(I, \mathcal{R})$ to $\text{chase}_1(I, \mathcal{R}')$, $\text{chase}_\infty(I, \mathcal{R})$ is finite. Now, for each rule $R \in \mathcal{R}'$, let $I = \text{freeze}(\text{body}(\text{R}))$: since R is applicable to I and $\text{chase}_1(I, \mathcal{R}')$ maps to $\text{chase}_\infty(I, \mathcal{R})$, there is k_R such that $\text{head}(\text{R})$ maps to $\text{chase}_{k_R}(I, \mathcal{R})$. Let k be the maximum k_R over all rules $R \in \mathcal{R}$. For any instance I, there is n such that $\text{chase}_\infty(I, \mathcal{R}) = \text{chase}_n(I, \mathcal{R})$ with $n \leq k$. Hence, $\mathcal{R}$ is bounded (by k). □

The converse is however not true, as witnessed by the following example.

Example 5 (Prime Example)**.** *Let $\mathcal{R} = \{R_1, R_2\}$ where:*

$R_1 : A(x) \to \exists z\, p(x, z)$
$R_2 : p(x, z) \wedge B(y) \to r(z, y)$

$\mathcal{R}$ is bounded but not parallelisable. Let, for any n, $I_n = \{A(a), B(b_1), \ldots, B(b_n)\}$. There is a null in $\text{chase}_\infty(I_n, \mathcal{R})$ that appears in $n+1$ atoms (apply R_1 once, which creates a null, then apply R_2 n times). Hence, there is no finite set of rules $\mathcal{R}'$ such that $chase_\infty(I_n, \mathcal{R})$ injectively maps to $chase_1(I, \mathcal{R}')$ for any n.

Motivated by this example, we now introduce a new class of rules, called *pieceful*.

3.2 The Pieceful Class

In short, pieceful rule sets ensure that for any rule application, the entire rule frontier is mapped either to terms from the initial instance or to terms that occur in atoms brought by a single previous rule application.

Definition 3 (Pieceful Derivation)**.** *An $\mathcal{R}$-derivation $(I_0 = I), \ldots, I_k$, is* pieceful *if for all i with $0 < i \leq k$, either $\pi_i(\text{fr}(R_i)) \subseteq \text{term}(I)$ or there is $j < i$ such that $\pi_i(\text{fr}(R_i)) \subseteq \text{term}(A_j)$, where $A_j = \pi_j^{\text{safe}}(\text{head}(\text{R}_\text{j}))$.*

Definition 4 (Pieceful Rule Set)**.** *A rule set $\mathcal{R}$ is* pieceful *if (for any instance I) any $\mathcal{R}$-derivation (from I) is pieceful.*

Example 6. *Consider again Example 5. From $I = \{A(a), B(b)\}$, one builds a non-pieceful derivation $(I_0 = I)(R_1, \pi_1, I_1)(R_2, \pi_2, I_2)$. Indeed, (R_1, π_1) produces $\pi_1^{\text{safe}}(\text{head}(\text{R}_1)) = A_1 = \{p(a, \nu_0)\}$, with ν_0 the null created from z. Then, $\pi_2 = \{x \mapsto a, y \mapsto b, z \mapsto \nu_0\}$. Since $\text{fr}(R_2) = \{y, z\}$ is mapped to $\{b, \nu_0\}$, with $b \notin \text{term}(A_1)$ and $\nu_0 \notin \text{term}(I)$, this derivation is not pieceful, hence neither is $\mathcal{R}$.*

Interestingly, when a rule set is not pieceful, it is possible to build instances that generalize the situation from Example 5, so that the chase creates nulls that occur in an arbitrarily large number of atoms, as shown by next Proposition 2. As will become clear later (Proposition 5), the reciprocal statement is true when the rule set is chase-finite.

Proposition 2. *If $\mathcal{R}$ is not pieceful then, for all n, there exist an instance I_n and a null ν_n such that ν_n occurs in at least n atoms in $chase_\infty(I_n, \mathcal{R})$.*

Proof. (Sketch) Let $I^0, \ldots, I^{n-1}, I^n$ be a non-pieceful derivation such that $I^0, \ldots, I^{n-1}$ is pieceful, *i.e.*, (R_n, π_n) applied on I^{n-1} is the first application that violates the pieceful condition. From that derivation, we build a set of instances $\{I_i\}$ such that the chase of I_i contains a null that occurs in at least i (distinct) atoms.

At least one frontier variable of R_n is mapped to a null. Let k be the largest integer such that (R_k, π_k) introduces a null (in I^k) to which a frontier variable of R_n is mapped by π_n. Let ν^* be this null. We define by induction on i the instance I_i as follows, where each $f_i(I^{k-1})$ denotes a freezing of I^{k-1}: *(i)* $I_0 = f_0(I^{k-1})$ and *(ii)* for any $i \geq 1$, $I_i = I_{i-1} \cup f_i(I^{k-1})$ where: if $x \in \pi_k(\mathrm{fr}(R_k))$, then $f_i(x) = f_0(x)$, otherwise $f_i(x)$ is a fresh constant (w.r.t. the whole construction). Intuitively, I_i is built from $i+1$ copies of I^{k-1}, where all the terms have been freshly renamed except for those in $\pi_k(\mathrm{fr}(R_k))$. From any I_i, we can build a derivation that mimics the initial derivation $I^{k-1}, \ldots, I^n$ and show that ν occurs in $i+1$ atoms. □

It follows that any parallelisable set is pieceful:

Proposition 3. *If $\mathcal{R}$ is parallelisable, then it is pieceful.*

Proof. Assume that $\mathcal{R}$ is parallelisable by $\mathcal{R}'$. Notice that for any I, any null in $\mathrm{chase}_\infty(I, \mathcal{R})$ occurs in at most h atoms, where h is the maximal size of a rule head in $\mathcal{R}'$ (indeed, two distinct rule applications from $\mathcal{R}'$ cannot share any null). Since h is bounded independently from I, by contraposite of Proposition 2, $\mathcal{R}$ is pieceful. □

How does the pieceful class fit in the existential rule landscape? Clearly, pieceful rule sets are *greedy-bounded-treewidth sets (gbts)* (Baget et al. 2011b; Rudolph et al. 2014)[1], an expressive family for which CQ answering is decidable. The gbts class includes some prominent existential classes, see Figure 1. There are three basic classes: (plain) *datalog* (e.g., (Abiteboul, Hull, and Vianu 1994)), in which there are no existential variables at all; *guarded* rules, in which all the variables from a rule body are guarded, *i.e.*, jointly occur in a body atom (Calì, Gottlob, and Lukasiewicz 2009; Calì, Gottlob, and Kifer 2013); *frontier-one* rules, in which the frontier of a rule is restricted to (at most) one variable (Baget et al. 2009). Combining guardedness and frontier-based restrictions leads to *frontier-guarded* rules, in which only the frontier of a rule needs to be guarded (Baget, Leclère, and Mugnier 2010). The guardedness condition is further relaxed in *weakly (frontier) guarded (w(f)g)* rules, in which only the (frontier) variables possibly mapped to nulls during the chase need to be guarded.

[1]Indeed, a pieceful derivation is a specific greedy derivation and gbts are those sets for which all derivations are greedy.

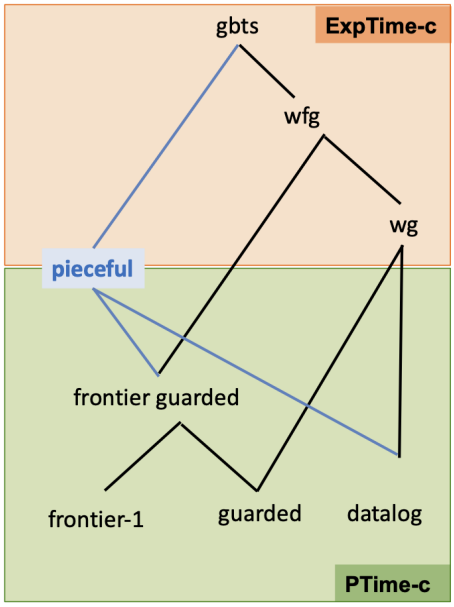

Figure 1: The gbts classes ordered by syntactic inclusion (and the data complexity of the associated Boolean CQ answering problem)

As pictured in Figure 1, the pieceful class includes datalog and frontier-guarded, but not wg (Example 5 is wg) and it is not difficult to see that it is actually incomparable with wfg.

Proposition 4. *Any set of frontier-guarded existential rules or of datalog rules is pieceful.*

3.3 Characterizing Parallelisability

We first point out that the chase of any instance I is equal to the union of its *pieces* (w.r.t. nulls). A fundamental property of pieceful rules is that the application of a rule $B \rightarrow H$ at level i can *never connect two pieces* of $\mathrm{chase}_{i-1}(I, \mathcal{R})$; when it does not produce an atom already present in $\mathrm{chase}_{i-1}(I, \mathcal{R})$, either it creates a new piece of size $|H|$ (when the frontier is mapped to terms from I), or it makes an existing piece grow by $|H|$ atoms (when the frontier is mapped to terms with at least one null).

Proposition 5. *If $\mathcal{R}$ is pieceful then for any instance I and integer k, the maximal size of a piece in $\mathrm{chase}_k(I, \mathcal{R})$ is bounded independently from I.*

Proof. (Sketch) Given I and $\mathcal{R}$, we note $P(i)$ the maximal size of a piece in $\mathrm{chase}_i(I, \mathcal{R})$. We prove that $P(0) = 1$ and, for $i \geq 0$, $P(i+1) \leq (P(i) \times a)^{\mathrm{fr}} \times h \times |\mathcal{R}|$, where a is the maximal arity of a predicate, fr and h are the maximal size of a rule frontier in $\mathcal{R}$ and a rule head in $\mathcal{R}$, respectively. □

Corollary 1. *If $\mathcal{R}$ is pieceful and bounded, then for any instance I, the maximal size of a piece in $\mathrm{chase}_\infty(I, \mathcal{R})$ is bounded independently from I.*

Based on this corollary, we are now able to show that any bounded pieceful set is parallelisable. We will give another proof of this result in Sect. 4 (see Cor. 2).

Proposition 6. *If $\mathcal{R}$ is pieceful and bounded then it is parallelisable.*

Proof. (Sketch) As $\mathcal{R}$ is pieceful and bounded, there is a finite set of pieces $\mathcal{P}$ such that for any instance I, any piece of $\mathrm{chase}(I, \mathcal{R})$ is isomorphic to a piece of $\mathcal{P}$. By isomorphism from a piece P_1 to a piece P_2, we mean a bijection b from $\mathrm{term}(P_1)$ to $\mathrm{term}(P_2)$ such that, for all $x \in \mathrm{term}(P_1)$, $b(x)$ is a null iff x is a null, and $b(P_1) = P_2$.

For any piece $P \in \mathcal{P}$, let $(\mathbf{c_P})$ be a tuple obtained by a total ordering on $\mathrm{const}(P)$, and consider the query $q_P(\mathbf{c_P}) = \exists \mathbf{y}\ P(\mathbf{c_P}, \mathbf{y})$, with $\mathbf{y}$ denoting the variables from P. Let Q_P be a finite complete set $\mathcal{R}$-rewritings of q_P (note that we need here the general definition of piece-unifiers that deals with constants). There is such set for any P, as $\mathcal{R}$ is bounded, hence *fus*. We define $\mathcal{R}'$ as the set of rules of the shape $q'_P(\mathbf{x}) \to \exists \mathbf{y} P(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ is a tuple of variables in bijection with $\mathbf{c_P}$, for $P \in \mathcal{P}$ and $q'_P \in Q_P$. □

From Propositions 1, 3 and 6, we finally obtain the following characterization of parallelisability:

Theorem 1. *A rule set is parallelisable if and only if it is both bounded and pieceful.*

4 Parallelisability and Rule Composition

Early work on datalog has shown that a datalog rule set $\mathcal{R}$ is 'parallelisable' (according to our definition) if and only if it is bounded (see, *e.g.*, (Gaifman et al. 1993)). Moreover, such set, say $\mathcal{R}^\star$, can be computed by an operation called *unfolding*: given rules $R_1 : B_1 \to H_1$ and $R_2 : B_2 \to H_2$, and a (most general) unifier u of an atom A in B_2 with the atom in H_1, the unfolding of R_2 by R_1 is the rule $u(B_1) \cup u(B_2 \setminus \{A\}) \to u(H_2)$; starting from $\mathcal{R}$, on can build $\mathcal{R}^\star$ by repeatedly unfolding a rule from $\mathcal{R}^\star$ by a rule from $\mathcal{R}$, until a fixpoint is reached. This is illustrated by the next example.

Example 7 (Datalog Unfolding)**.** *Let $\mathcal{R} = \{R_1, R_2, R_3\}$ with:*
$R_1 : A(x) \to B(x)$
$R_2 : C(x) \to D(x)$
$R_3 : B(x) \wedge D(x) \to G(x)$
Denoting $R_i \circ R_j$ the unfolding of R_i by R_j, we obtain:
$R_3 \circ R_1 : A(x) \wedge D(x) \to G(x)$
$R_3 \circ R_2 : C(x) \wedge B(x) \to G(x)$
$(R_3 \circ R_1) \circ R_2 : A(x) \wedge C(x) \to G(x)$
$(R_3 \circ R_2) \circ R_1 = (R_3 \circ R_1) \circ R_2$.
Finally, $\mathcal{R}^\star = \mathcal{R} \cup \{R_3 \circ R_1, R_3 \circ R_2, (R_3 \circ R_1) \circ R_2\}$. Here $\mathcal{R}^\star$ is finite, as $\mathcal{R}$ is bounded.

In the following, we will consider composition of existential rules, which generalizes this notion of unfolding.

4.1 Rule Composition

Composition of existential rules has been exploited as a means of tracing 'chained' sequences of rule applications (*e.g.*, (Baget et al. 2014; Wang, Wang, and Zhang 2018)). It is naturally based on the notions of piece-unifier and rewriting. The next definition of rule composition furthermore takes pieces into account.

Definition 5 (Rule Composition)**.** *Given rules $R_1 : B_1 \to H_1$ and $R_2 : B_2 \to H_2$ on disjoint sets of variables, and $\mu = (B'_2, H'_1, u)$ a piece-unifier of B_2 with R_1, the* (existential) composition *of R_2 with R_1 w.r.t. μ is the following existential rule, denoted by $R_2 \circ_\mu R_1$, or simply $R_2 \circ R_1$:*

1. *If $u(\mathrm{fr}(R_2)) \cap \mathrm{exist}(R_1) = \emptyset$:*
$$R_2 \circ_\mu R_1 = u(B_1) \cup u(B_2 \setminus B'_2) \to u(H_2)$$
2. *Otherwise:*
$$R_2 \circ_\mu R_1 = u(B_1) \cup u(B_2 \setminus B'_2) \to u(H_1) \cup u(H_2)$$

Note that $u(B_1) \cup u(B_2 \setminus B'_2)$ is exactly the rewriting of B_2 w.r.t. μ.

The first case in the definition is when no frontier variable from R_2 is unified with an existential variable from R_1, *i.e.*, $u(\mathrm{fr}(B'_2)) \subseteq \mathrm{fr}(R_1)$. Then, defining $R_2 \circ_\mu R_1$ as in Point 2 would lead to a rule with a two-piece head (resp., $u(H_1)$ and $u(H_2)$), which can be decomposed into two single-piece-head rules. Moreover, the rule $u(B_1) \cup u(B_2 \setminus B'_2) \to u(H_1)$ is useless because every application of this rule can be obtained with an application of R_1. In the second case, since at least one frontier variable from B'_2 is unified with an existential variable from R_1, $u(H_1) \cup u(H_2)$ forms a single piece. Therefore, restricting the rule head to $u(H_2)$ would result in a loss of information. Note that in both cases, the obtained rule $R_2 \circ R_1$ has a single piece head.

Example 8. *Let $\mathcal{R}$ contain three rules:*
$R_1 : A(x) \to \exists z\ p(x, z)$
$R_2 : p(x, z) \to B(z)$
$R_3 : C(x) \wedge B(y) \to r(x, y)$
The composed rule $R_3 \circ R_2 = p(x', z) \wedge C(x) \to r(x, z)$ illustrates Point 1. Defining $R_3 \circ R_2$ as in Point 2 would lead to the following rule with a two-piece head:
$p(x', z) \wedge C(x) \to B(z) \wedge r(x, z)$
We can see that $p(x', z) \wedge C(x) \to B(z)$ is useless w.r.t. R_2.
$R_2 \circ R_1 = A(x) \to \exists z\ p(x, z) \wedge B(z)$ *illustrates Point 2.*

The set $\mathcal{R}^\star$ includes the original set $\mathcal{R}$ as well as all rules obtained by composition.

Definition 6. *($\mathcal{R}^\star$) The set of (existentially-) composed rules associated with $\mathcal{R}$, denoted by $\mathcal{R}^\star$, is the possibly infinite set inductively defined as follows:*
(base) $\mathcal{R} \subseteq \mathcal{R}^\star$,
(induction) if $R_i, R_j \in \mathcal{R}^\star$ and there is a piece-unifier μ of $\mathrm{body(R_i)}$ with R_j, then $R_i \circ_\mu R_j \in \mathcal{R}^\star$.

$\mathcal{R}^\star$ is sound and complete in the sense that entailment of Boolean CQs is preserved (with $\mathcal{R}^\star$ applied in a single breadth-first step). Formally: for any rule set $\mathcal{R}$, instance I and Boolean CQ q, holds $I, \mathcal{R} \models q$ if and only if holds $\mathrm{chase}_1(I, \mathcal{R}^\star) \models q$. Soundness relies on the fact that all the rules in $\mathcal{R}^\star$ are entailed by $\mathcal{R}$ and completeness follows from (Wang, Wang, and Zhang 2018), Prop. 2.

The next proposition yields a more specific completeness result: if $\mathcal{R}$ is a pieceful rule set, then for any instance I, each piece of $chase_\infty(I, \mathcal{R})$ can be obtained by applying a rule from $\mathcal{R}^\star$ to I.

Proposition 7. *Let I be an instance and $\mathcal{R}$ be a set of rules. For any pieceful derivation of length i resulting in I_i, for any piece P in I_i, either $P \subseteq I$ or there exist a rule $R^* \in \mathcal{R}^\star$ and a homomorphism π from $\mathrm{body(R^*)}$ to I such that P maps to $\pi^{\mathrm{safe}}(\mathrm{head(R^*)})$ by an injective homomorphism.*

Proof. (Sketch) We prove the result by induction on the length i of the derivation. For $i = 1$, if $P \not\subseteq I$, then P has been generated by an application of a rule from $R \in \mathcal{R} \subseteq \mathcal{R}^\star$. Assuming that the result holds for any piece P of I_{i-1} with $i > 1$, we show it also holds for any piece P of I_i. Let $I_0, (R_1, \pi_1, I_1), \ldots, (R_i, \pi_i, I_i)$ be a derivation of length i. The body of R_i is mapped by π_i to k pieces of I_{i-1}, where $k \leq |\mathrm{var(body(R_i))}|$. Let P be the piece created or completed by the application of R_i by π_i. As the derivation is pieceful, P is either a piece of I_{i-1} to which $\pi_i^{\mathrm{safe}}(\mathrm{head(R_i)})$ has been added, or a new piece. We build by induction on k a rule R_P that maps by π_P to I and such that P injectively maps to the result of the application of R_P by π_P. □

We conjecture that the previous result actually holds without the pieceful restriction, but its proof would be more intricate. Thanks to this result, we can refine Proposition 6:

Corollary 2. *If $\mathcal{R}$ is pieceful and bounded then it is parallelisable by a (finite) subset of $\mathcal{R}^\star$.*

Proof. Since $\mathcal{R}$ is bounded, the maximum size of a piece is bounded and each piece can be generated by a derivation of length bounded by an integer n, where n is independent of the instance. □

Since CQ answering with general existential rules is not decidable, $\mathcal{R}^\star$ can be infinite. The next example shows that $\mathcal{R}^\star$ can be infinite even for $\mathcal{R}$ a *bounded* set of rules, which may seem surprising.

Example 9. *We consider again the prime example, where $\mathcal{R} = \{R_1, R_2\}$ is bounded. Let us build $\mathcal{R}^\star$:*
$R_1 : A(x) \rightarrow \exists z\, p(x, z)$
$R_2 : p(x, z) \wedge B(y) \rightarrow r(z, y)$
$R_2 \circ R_1 : A(x) \wedge B(y) \rightarrow \exists z\, p(x, z) \wedge r(z, y)$
$R_2 \circ (R_2 \circ R_1) : A(x) \wedge B(y) \wedge B(y_1) \rightarrow$
$\exists z\, p(x, z) \wedge r(z, y) \wedge r(z, y_1)$
etc. *At each step, one obtains a new rule by the composition $R_2 \circ R^*$, where R^* is the rule created at the preceding step:*
$A(x) \wedge B(y) \wedge B(y_1) \ldots B(y_i) \rightarrow$
$\exists z\, p(x, z) \wedge r(z, y) \wedge r(z, y_1) \ldots \wedge r(z, y_i)$
Not only $\mathcal{R}^\star$ is infinite, but no finite subset of it is complete.

Moreover, the previous example shows that rule compositions of the form $R \circ R^*$, with $R \in \mathcal{R}$ and $R^* \in \mathcal{R}^\star$ are required to achieve completeness, while rule compositions of the form $R^* \circ R$ are sufficient in the datalog fragment.

4.2 Existential Stability

In Sect. 3.3, we have shown that a rule set is parallelisable if and only if it is both bounded and pieceful, with this last notion being defined by the behavior of rules during the chase. The next question we study is whether pieceful rule sets can be characterized by their behavior during rule composition. We first introduce the 'existential stability' property (the reason for this name will be explained in Sect. 4.3).

Definition 7 (Existential stability)**.** *Given rules R_1 and R_2, we say that a piece-unifier $\mu = (B_2', H_1', u)$ of $\mathrm{body(R_2)}$ with R_1 satisfies the* (existential) stability *property if the following holds:*
- *either $u(\mathrm{fr}(R_2)) \cap \mathrm{exist}(R_1) = \emptyset$,*
- *or $\mathrm{fr}(R_2) \subseteq \mathrm{var}(B_2')$.*

This notion is extended to a rule set: $\mathcal{R}$ satisfies the stability property if, for any rules R_1 and R_2 in $\mathcal{R}$, any piece-unifier of R_2 with R_1 satisfies the stability property.

Informally, the stability property says that when a frontier variable from R_2 is unified with an existential variable from R_1 then all the frontier variables from R_2 are unified. By the next example, we point out that the stability property of a rule set may not be preserved when composed rules are added.

Example 10. *Let $\mathcal{R}$ from Example 8. It can be checked that it has the stability property. Now, consider $R_2 \circ R_1$ and R_3:*
$R_2 \circ R_1 : A(x) \rightarrow \exists z\, p(x, z) \wedge B(z)$
$R_3 : C(x) \wedge B(y) \rightarrow r(x, y)$
Then, $R_3 \circ (R_2 \circ R_1)$ involves a piece-unifier that does not satisfy the stability property: $\mathrm{fr}(R_3) = \{x, y\}$; y is unified with $z \in \mathrm{exist}(R_2 \circ R_1)$ but x is not unified, hence $\mathcal{R} \cup \{R_2 \circ R_1\}$ does not have the stability property.

We say that a rule set $\mathcal{R}$ is *stable at the infinite* if $\mathcal{R}^\star$ satisfies the stability property. Next, we show that pieceful rule sets are exactly those rule sets stable at the infinite.

Proposition 8. *Any pieceful rule set satisfies the stability property.*

Proof. Let $R_1 : B_1 \rightarrow H_1$ and $R_2 : B_2 \rightarrow H_2$ be two rules of a pieceful rule set and let $\mu = (B_2', H_1', u)$ be a piece-unifier of $\mathrm{body(R_2)}$ with R_1. Assume that μ does not satisfy the stability property. Let $I = u(B_1) \cup u(B_2 \setminus B_2')$. For simplicity here, we confuse I and its freezing, and we assume that safe renamings are the identity (indeed, we will consider a single application of R_1 followed by a single application of R_2). R_1 is applicable on I by a homomorphism h_1 extending $u_{|\mathrm{var}(B_1)}$ (*i.e.*, for all $x \in \mathrm{var}(B_1)$, $h_1(x) = u(x)$ if $x \in \mathrm{fr}(R_1)$, otherwise $h_1(x) = x$). R_2 is applicable on the result of this application, *i.e.*, $I \cup h_1(H_1)$, by a homomorphism h_2 extending $u_{|\mathrm{var}(B_2)}$ (*i.e.*, for all $x \in \mathrm{var}(B_2)$, $h_2(x) = u(x)$ if $u(x)$ is defined, otherwise $h_2(x) = x$). Since u does not satisfy the stability property, for some $x \in \mathrm{fr}(R_2)$, $u(x) \in \mathrm{exist}(R_1)$ and for another $x' \in \mathrm{fr}(R_2)$, $u(x')$ is not defined. Hence, $h_2(x) \notin \mathrm{term}(I)$, while $h_2(x') \notin \mathrm{term}(h_1(H_1))$, which shows that the derivation is not pieceful. □

Proposition 9. *If $\mathcal{R}$ is pieceful, then for any rules R_1 and R_2 from $\mathcal{R}$, $\mathcal{R} \cup \{R_2 \circ_\mu R_1\}$ is also pieceful.*

Proof. (Sketch) We show the contrapositive. Assume $\mathcal{R}' = \mathcal{R} \cup \{R_2 \circ_\mu R_1\}$ is not pieceful. Let $D' = I_0', \ldots, I_n'$ be a pieceful $\mathcal{R}'$-derivation on which (R_{n+1}, π_{n+1}) is applicable in a non-pieceful way. We build a non-pieceful $\mathcal{R}$-derivation $D = I_1, \ldots, I_{\varphi(n)}$ by induction on n, which satisfies the following:

- For all $1 \leq i \leq n$, there is an isomorphism ψ_i from I'_i to $I_{\varphi(i)}$
- For any set A_j (produced by an application in D), there is A'_k (produced by an application in D') s.t. $A_j \subseteq \psi_n(A'_k)$.

Consider now (R_{n+1}, π_{n+1}). Then $\psi_n \circ \pi_{n+1}$ is a homomorphism from $\mathrm{body(R_{n+1})}$ to $I_{\varphi(n)}$. If $R_{n+1} \in \mathcal{R}$, by hypothesis on (R_{n+1}, π_{n+1}), there is no A'_k s.t. $\pi_{n+1}(\mathrm{fr}(R_{n+1}))) \subseteq \mathrm{term}(A'_k)$. By induction hypothesis, for every A_j in D, there is $A'_{k'}$ s.t. $A_j \subseteq \psi_n(A'_{k'})$. Hence, there is no A_j in D s.t. $\psi_n \circ \pi_{n+1}(\mathrm{fr}(R_{n+1})) \subseteq \mathrm{term}(A_j)$, and $\mathcal{R}$ is not pieceful. If $R_{n+1} = R_2 \circ_\mu R_1$, the same reasoning applies, noting that, since μ satisfies the stability property, either $\mathrm{fr}(R_{n+1}) \subseteq u(\mathrm{fr}(R_2))$ or $\mathrm{fr}(R_{n+1}) \subseteq u(\mathrm{fr}(R_1))$. □

From Prop. 8 and 9, we obtain that pieceful rule sets are stable at the infinite. We now show the converse direction.

Proposition 10. *Any rule set that is stable at the infinite is pieceful.*

Proof. (Sketch) We prove the result by contrapositive. Considering the first application (R, π) that violates the pieceful constraint in a derivation, we consider a piece P that contains $\pi(x)$ for some $x \in \mathrm{fr}(R)$. We consider R_P that generates P (using Proposition 7), and build a piece-unifier μ of $\mathrm{body(R)}$ with R_P that violates the stability property. □

From the three previous propositions, we obtain the desired result:

Theorem 2. *A rule set is pieceful if and only if it is stable at the infinite.*

Finally, as a corollary of Th. 1 and 2, we obtain another characterization of parallelisable sets of rules.

Corollary 3. *A rule set is parallelisable if and only if it is both bounded and stable at the infinite.*

4.3 Beyond Existential Composition

In this section, we question the notion of rule composition and provide preliminary findings. We define another rule composition operation, which seems to fit better with our intuition and is more succinct (hence, its name 'compact'), but which goes beyond existential rules. We then show that, for piece-unifiers that satisfy the stability property, existential and compact compositions actually coincide.

For datalog, we know that $\mathcal{R}^\star$ is finite if and only if $\mathcal{R}$ is bounded. To better understand why this is no longer true for existential rules, let us focus on compositions of the form $R_2 \circ R_1$, with $\mathrm{exist}(R_1) \neq \emptyset$. Intuitively, the rules $R_2 \circ R_1$ capture the situations in the chase where 'an application of R_1 leads to trigger a new application of R_2'. More formally: for any instance I, application of R_1 to I yielding I_1, homomorphism π_2 from $\mathrm{body(R_2)}$ to I_1 such that $\pi_2(\mathrm{body(R_2)}) \not\subseteq I$, with $I_2 = \alpha(I_1, R_2, \pi_2)$, there is a composed rule $R_2 \circ R_1$ whose application to I yields an instance isomorphic to I_2. One might be tempted to conclude that the set of all rules of the form $R_2 \circ_\mu R_1$ (*i.e.*, for all piece-unifiers μ) is able to capture 'all applications of R_2 that use an atom brought by an application of R_1', *i.e.*, $\mathrm{chase}_1(I, \{R_1, R_2 \circ_\mu R_1 | \forall \mu\})$ would be equivalent to $I_1 = \mathrm{chase}_1(I, \{R_1\})) \cup \{\alpha(I_1, R_2, \pi_2) | \pi_2(\mathrm{body(R_2)}) \not\subseteq I\})$.

However, this does not hold, as illustrated below.

Example 11. *Consider again the prime example with $\mathcal{R} = \{R_1, R_2\}$:*
$R_1 : A(x) \rightarrow \exists z\, p(x, z)$
$R_2 : p(x, z) \wedge B(y) \rightarrow r(z, y)$
$R_2 \circ R_1 : A(x) \wedge B(y) \rightarrow \exists z\, p(x, z) \wedge r(z, y)$.
Let $I = \{A(a), B(b), B(c)\}$. Then:

$$chase_\infty(I, \mathcal{R}) = I \cup \{p(a, z_0), r(z_0, b), r(z_0, c)\}$$

As it is obtained by one application of R_1 which triggers two parallel applications of R_2, this also corresponds to the above $(I_1 = chase_1(I, \{R_1\})) \cup \{\alpha(I_1, R_2, \pi_2) | \pi_2(\mathrm{body(R_2)}) \not\subseteq I\}$. One (breadth-first) chase step with $\mathcal{R} \cup \{R_2 \circ R_1\}$ would produce instead:

$$I \cup \{p(a, z_0), p(a, z_1), r(z_1, b), p(a, z_2), r(z_2, c)\}$$

Note that this is here equal to $chase_1(I, \{R_1, R_2 \circ_\mu R_1 | \forall \mu\})$. We can see that two nulls z_1 and z_2 are created instead of a single one. Obviously, both results are not equivalent. F.i., the Boolean CQ $q() = \exists u\, r(u, b) \wedge r(u, c)$ would be answered positively in the first case, but not in the second.

In the previous example, the logical formula associated with $R_2 \circ R_1$ is the following:

$$\forall x \forall y\ (A(x) \wedge B(y) \rightarrow \exists z\ (p(x, z) \wedge r(z, y))).$$

Instead, we propose to interpret rule composition as:

$$\forall x\ (A(x) \rightarrow \exists z\ (p(x, z) \wedge \forall y\ (B(y) \rightarrow r(z, y))))$$

By removing the knowledge entailed by R_1, we obtain:

$$R_2 \bullet R_1 = \forall x \exists z \forall y\ (A(x) \wedge B(y) \rightarrow p(x, z) \wedge r(z, y))$$

Definition 8 (Compact Composition)**.** *Let*
$R_1 = \forall \mathbf{x_1} \forall \mathbf{y_1}\ [\ B_1(\mathbf{x_1}, \mathbf{y_1}) \rightarrow \exists \mathbf{z_1}\ H_1(\mathbf{x_1}, \mathbf{z_1})\ ]$ *and*
$R_2 = \forall \mathbf{x_2} \forall \mathbf{y_2}\ [\ B_2(\mathbf{x_2}, \mathbf{y_2}) \rightarrow \exists \mathbf{z_2}\ H_2(\mathbf{x_2}, \mathbf{z_2})\ ]$.
Let $\mu = (B'_2, H'_1, u)$ be a piece-unifier of B_2 with R_1.

The compact composition *of R_2 with R_1 w.r.t. μ, denoted by $R_2 \bullet_\mu R_1$ is the following closed formula:*

$$\forall \mathbf{x'_1} \forall \mathbf{y_1} \exists \mathbf{z_1} \forall \mathbf{x'_2} \forall \mathbf{y'_2}$$
$$(u(B_1) \wedge u(B_2 \setminus B'_2) \rightarrow \exists \mathbf{z_2}\ (u(H_1) \wedge u(H_2)))$$

where $\mathbf{x'_1} = u(\mathbf{x_1})$, $\mathbf{x'_2} = \mathbf{x_2} \setminus \mathrm{var}(B'_2)$, $\mathbf{y'_2} = \mathbf{y_2} \setminus \mathrm{var}(B'_2)$.

Compact composition is more succinct than existential composition, since a single $\bullet$-composed rule may capture an unbounded number of $\circ$-composed rules. However, the resulting formula is generally not an existential rule. We show below that, when the piece-unifiers involved in rule composition satisfy the stability property, the result of compact composition is equivalent to an existential rule (hence the name 'existential stability') and it coincides with existential composition (for clarity, we consider below the general form of existential composition, *i.e.*, ignore Point 1 in Def. 5).

Proposition 11. *When the stability property is satisfied, the compact composition is equivalent to the existential composition,* i.e.*, for all rules R_1 and R_2, and any piece-unifier μ of* $\mathrm{body}(\mathrm{R}_2)$ *with R_1 satisfying the stability property, $R_2 \circ_\mu R_1 \equiv R_2 \bullet_\mu R_1$ (where $\equiv$ denotes the logical equivalence and $\circ$ is defined according to Point 2 of Def. 5).*

Proof. Note that $R_2 \circ_\mu R_1$ is equivalent to the formula obtained from $R_2 \bullet_\mu R_1$ by moving $\exists \mathbf{z_1}$ after $\forall \mathbf{x'_2} \forall \mathbf{y'_2}$. Here, such an inversion of quantifiers can be done without change of semantics if we can group the atoms in $R_2 \bullet_\mu R_1$ into two sets, such that one contains all the atoms with variables in $\mathbf{z_1}$ and the other all the atoms with variables in $\mathbf{x'_2} \cup \mathbf{y'_2}$. Since $u(H_2)$ may contain variables from both $\mathbf{x'_2}$ and $\mathbf{z_1}$, this is possible if and only if $u(H_2)$ contains no variable from $\mathbf{x'_2}$ or no variable from $\mathbf{z_1}$. We check that it is indeed the case when μ has the stability property: if $u(\mathrm{fr}(R_2)) \cap \mathrm{exist}(R_1) = \emptyset$, then $u(H_2)$ does not contain any variable from $\mathbf{z_1}$; if $\mathrm{fr}(R_2) \subseteq \mathrm{var}(B'_2)$, then $\mathbf{x'_2}$ is empty, hence $u(H_2)$ does not contain any variable from $\mathbf{x'_2}$. □

Finally, one could think of skolemizing existential rules to get (specific) logic-programming rules. Briefly, skolemization consists of replacing each existential variable in a rule head by a fresh functional term over the rule frontier. Then, rule composition is based on classical (most general) unifiers. However, as illustrated below, the composition of two skolemized existential rules may not be a skolemized existential rule. Actually, such composition can be seen as the skolemization of the formula obtained by compact composition, which leads us again beyond the existential rule fragment.

Example 12 (Skolem composition)**.** *Consider the skolemization (noted sk) of the rules from the prime example:*
$sk(R_1) : A(x) \rightarrow p(x, f(x))$
$sk(R_2) = R_2 : p(x, z) \wedge B(y) \rightarrow r(z, y)$
Then, the composition of $sk(R_2)$ with $sk(R_1)$ yields the following rule, where $p(x, f(x))$ could be removed:

$$A(x) \wedge B(y) \rightarrow p(x, f(x)) \wedge r(f(x), y)$$

This rule is not the skolemization of any existential rule because f does not span over the whole rule frontier $\{x, y\}$. Instead, it is $sk(R_2 \bullet R_1)$.

5 Concluding Remarks

In this paper, we introduce the notion of parallelisability of a set of existential rules and characterize parallelisable rule sets in two different ways. One characterization relies on the behavior of rules during the chase, which leads to define a new class of existential rules, namely pieceful. Another characterization relies on the behavior of rules during rewriting, which led us to question the notion of rule composition. We believe that these results open up many perspectives, which we now outline.

Application to OBDA Mature systems, such as OnTop (Calvanese et al. 2017) or Mastro (Calvanese et al. 2011), are based on lightweight description logics (typically DL-Lite$_\mathcal{R}$ underpinning the W3C language OWL2 QL) and the mapping is GAV (*i.e.*, mapping assertions $q_S(\mathbf{x}) \rightarrow q_O(\mathbf{x})$ satisfy $\mathrm{var}(q_O) \subseteq \mathbf{x}$). Some other OBDA systems are based on the lightweight ontological language RDFS, *e.g.*, UltraWrap (Sequeda, Arenas, and Miranker 2014), where RDFS is extended with inverse and transitive properties, and mappings are still GAV, and Obi-Wan (Buron et al. 2020a), strictly restricted to RDFS but with GLAV mappings (i.e., q_O is any CQ).We believe that the existential rule framework is particularly well suited to the development of OBDA systems in more expressive settings. Indeed, existential rules are able to express both ontological knowledge and relational GLAV mappings; GLAV mappings provide increased flexibility compared to GAV mappings, by their ability to invent values, thanks to existential variables. This yields a uniform setting for the whole OBDA specification, thereby facilitating the analysis of the interactions between $\mathcal{O}$ and $\mathcal{M}$, which is key to efficiency. Furthermore, a rich family of existential rule dialects achieving various expressivity/tractability tradeoffs are available. In this paper, we have taken a first step towards the design of compilation techniques for this setting, by characterizing the notion of parallelisability. Indeed, when the rule set is parallelisable, ontological reasoning can be totally compiled into the mapping. Our results pave the way for the development of query answering techniques exploiting parallelisation and the specificities of existential rule dialects. Another interesting extension of our work would be to investigate combined reasoning (Lutz, Toman, and Wolter 2009) or partitioned reasoning (Baget et al. 2011a), which would allow us to go beyond parallelisable rule sets.

Deepening Theoretical Foundations The novel pieceful class is of interest in itself, specially in the context of query answering. It includes both datalog and frontier-guarded, a prominent class of existential rules. Frontier-guarded itself covers some major DL dialects used in query answering (*e.g.*, DL-Lite$_\mathcal{R}$ or $\mathcal{ELHI}$, see (Mugnier 2020) for more details on the relationships between these DLs and the gbts family). We conjecture that CQ answering with pieceful rules is in PTime for data complexity. By all these features, the pieceful class is related to the recently introduced warded class (Bellomarini, Sallinger, and Gottlob 2018), even if both are incomparable, at least from a syntactic viewpoint (our prime example is warded but not pieceful, on the other hand warded does not include frontier-guarded).

Concerning the notion of parallelisability itself, we based our study on the semi-oblivious chase. In line with this, parallelisability requires an *injective* homomorphism from $\mathrm{chase}_\infty(I, \mathcal{R})$ to $\mathrm{chase}_1(I, \mathcal{R}')$ (where $\mathcal{R}'$ is the parallelisation of $\mathcal{R}$). Dropping the injectivity requirement, *i.e.*, considering logical equivalence, would lead to a more general notion of parallelisability, which remains to be investigated.

Finally, we have seen that compact rule composition leads us beyond the existential rule fragment. The obtained formulas belong to strictly more expressive logical fragments that have been studied in particular in (Gottlob, Pichler, and Sallinger 2015). Whether such fragments could lead to parallelisation techniques is another open issue.

Acknowledgements

This work is partly supported by the ANR project CQFD (ANR-18-CE23-0003).

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1994. *Foundations of Databases*. Addison Wesley.

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

Baget, J.-F.; Leclère, M.; Mugnier, M.-L.; and Salvat, E. 2009. Extending Decidable Cases for Rules with Existential Variables. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI 2009*, 677–682.

Baget, J.-F.; Leclère, M.; Mugnier, M.-L.; and Salvat, E. 2011a. On rules with existential variables: Walking the decidability line. *Artificial Intelligence* 175(9-10):1620–1654.

Baget, J.-F.; Mugnier, M.-L.; Rudolph, S.; and Thomazo, M. 2011b. Walking the Complexity Lines for Generalized Guarded Existential Rules. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI 2011*, 712–717.

Baget, J.; Garreau, F.; Mugnier, M.; and Rocher, S. 2014. Extending acyclicity notions for existential rules. In Schaub, T.; Friedrich, G.; and O'Sullivan, B., eds., *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, 39–44. IOS Press.

Baget, J.-F.; Leclère, M.; and Mugnier, M.-L. 2010. Walking the Decidability Line for Rules with Existential Variables. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010*. AAAI Press.

Bellomarini, L.; Sallinger, E.; and Gottlob, G. 2018. The vadalog system: Datalog-based reasoning for knowledge graphs. *Proc. VLDB Endow.* 11(9):975–987.

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

Bourhis, P.; Leclère, M.; Mugnier, M.; Tison, S.; Ulliana, F.; and Gallois, L. 2019. Oblivious and semi-oblivious boundedness for existential rules. In Kraus, S., ed., *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, 1581–1587. ijcai.org.

Buron, M.; Goasdoué, F.; Manolescu, I.; and Mugnier, M. 2020a. Obi-wan: Ontology-based RDF integration of heterogeneous data. *Proc. VLDB Endow.* 13(12):2933–2936.

Buron, M.; Goasdoué, F.; Manolescu, I.; and Mugnier, M. 2020b. Ontology-based RDF integration of heterogeneous data. In *Proceedings of the 23nd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020*, 299–310.

Buron, M.; Mugnier, M.-L.; and Thomazo, M. 2021. Parallelisable existential rules: a story of pieces. *CoRR* abs/2107.06054.

Calì, A.; Gottlob, G.; and Kifer, M. 2013. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res. (JAIR)* 48:115–174.

Calì, A.; Gottlob, G.; and Lukasiewicz, T. 2009. A General Datalog-Based Framework for Tractable Query Answering over Ontologies. In *Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009*, 77–86. ACM.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family. *Journal of Automated Reasoning* 39(3):385–429.

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).

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).

Delivorias, S.; Leclère, M.; Mugnier, M.; and Ulliana, F. 2021. Characterizing boundedness in chase variants. *Theory Pract. Log. Program.* 21(1):51–79.

Deutsch, A.; Nash, A.; and Remmel, J. 2008. The chase revisited. In *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008*, 149–158.

Gaifman, H.; Mairson, H. G.; Sagiv, Y.; and Vardi, M. Y. 1993. Undecidable optimization problems for database logic programs. *J. ACM* 40(3):683–713.

Gottlob, G.; Pichler, R.; and Sallinger, E. 2015. Function symbols in tuple-generating dependencies: Expressive power and computability. In Milo, T., and Calvanese, D., eds., *Proceedings of the 34th ACM Symposium on Principles of Database Systems, PODS 2015, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, 65–77. ACM.

Hillebrand, G. G.; Kanellakis, P. C.; Mairson, H. G.; and Vardi, M. Y. 1995. Undecidable boundedness problems for datalog programs. *J. Log. Program.* 25(2):163–190.

König, M.; Leclère, M.; Mugnier, M.; and Thomazo, M. 2015. Sound, complete and minimal ucq-rewritings for existential rules. *Semantic Web Journal* 6(5):451–475.

Kontchakov, R.; Rezk, M.; Rodriguez-Muro, M.; Xiao, G.; and Zakharyaschev, M. 2014. Answering SPARQL queries over databases under OWL 2 QL entailment regime. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference*, volume 8796, 552–567. Springer.

Lenzerini, M. 2018. Managing data through the lens of an ontology. *AI Mag.* 39(2):65–74.

Lutz, C.; Toman, D.; and Wolter, F. 2009. Conjunctive Query Answering in the Description Logic $\mathcal{EL}$ Using a Relational Database System. In *Proceedings of the 21st Inter-*

national Joint Conference on Artificial Intelligence, IJCAI 2009, 2070–2075.

Marnette, B. 2009. Generalized schema-mappings: from termination to tractability. In *Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009*, 13–22.

Mugnier, M. 2020. Data access with horn ontologies: Where description logics meet existential rules. *Künstliche Intell.* 34(4):475–489.

Poggi, A.; Lembo, D.; Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Rosati, R. 2008. Linking data to ontologies. *J. Data Semantics* 10:133–173.

Rudolph, S.; Thomazo, M.; Baget, J.-F.; and Mugnier, M.-L. 2014. Worst-case optimal query answering for greedy sets of existential rules and their subclasses. *CoRR* abs/1412.4485.

Sequeda, J. F.; Arenas, M.; and Miranker, D. P. 2014. OBDA: query rewriting or materialization? in practice, both! In *ISWC*.

Wang, Z.; Wang, K.; and Zhang, X. 2018. Forgetting and unfolding for existential rules. In McIlraith, S. A., and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2013–2020. AAAI Press.

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. ijcai.org.