TLP **17** (5–6): 726–743, 2017. © Cambridge University Press 2017
doi:10.1017/S1471068417000369 First published online 24 August 2017

*Finite model reasoning over existential rules**

GIOVANNI AMENDOLA, NICOLA LEONE and MARCO MANNA
Department of Mathematics and Computer Science, University of Calabria, Via Pietro Bucci, 87036 Arcavacata, Rende, Cosenza, Italy
(*e-mails:* amendola@mat.unical.it,leone@mat.unical.it,manna@mat.unical.it)

submitted 2 May 2017; revised 20 June 2017; accepted 4 July 2017

Abstract

Ontology-based query answering asks whether a Boolean conjunctive query is satisfied by all models of a logical theory consisting of a relational database paired with an ontology. The introduction of existential rules (i.e., Datalog rules extended with existential quantifiers in rule heads) as a means to specify the ontology gave birth to Datalog+/−, a framework that has received increasing attention in the last decade, with focus also on decidability and finite controllability to support effective reasoning. Five basic decidable fragments have been singled out: linear, weakly acyclic, guarded, sticky, and shy. Moreover, for all these fragments, except shy, the important property of finite controllability has been proved, ensuring that a query is satisfied by all models of the theory iff it is satisfied by all its finite models. In this paper, we complete the picture by demonstrating that finite controllability of ontology-based query answering holds also for shy ontologies, and it therefore applies to all basic decidable Datalog+/− classes. To make the demonstration, we devise a general technique to facilitate the process of (dis)proving finite controllability of an arbitrary ontological fragment.

KEYWORDS: existential rules, Datalog, finite controllability, finite model reasoning, query answering.

1 Introduction

The problem of answering a Boolean query q against a logical theory consisting of an extensional database D paired with an ontology Σ is attracting the increasing attention of scientists in various fields of Computer Science, ranging from Artificial Intelligence (Baget *et al.* 2011; Bourhis *et al.* 2016). This problem, called *ontology-based query answering*, for short OBQA (Calì *et al.* 2009b), is usually stated as $D \cup \Sigma \models q$, and it is equivalent to checking whether q is satisfied by all models of $D \cup \Sigma$ according to the standard approach of first-order logics, yielding an open world semantics.

* The paper has been partially supported by the Italian Ministry for Economic Development (MISE) under project "PIUCultura – Paradigmi Innovativi per l'Utilizzo della Cultura" (n. F/020016/01-02/X27), and under project "Smarter Solutions in the Big Data World (S2BDW)" (n. F/050389/01-03/X32) funded within the call "HORIZON2020" PON I&C 2014-2020.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

Description logics (Baader *et al.* 2003) and Datalog$^{\pm}$ (Calì *et al.* 2009a) have been recognized as the two main families of formal knowledge representation languages to specify Σ, while union of (Boolean) conjunctive queries is the most common and studied formalism to express q. For both these families, OBQA is generally undecidable (Rosati 2007; Calì *et al.* 2013). Hence, a number of syntactic decidable fragments of the above ontological languages have been singled out. However, decidability alone is not the only desideratum. For example, a good balance between computational complexity and expressive power is, without any doubt, of high importance too. But there is another property that is turning out to be as interesting as challenging to prove: it goes under the name of *finite controllability* (Rosati 2006). An ontological fragment $\mathscr{F}$ is finitely controllable if, for each triple $\langle D, \Sigma, q \rangle$ with $\Sigma \in \mathscr{F}$, it holds that $D \cup \Sigma \not\models q$ implies that there is a finite model M of $D \cup \Sigma$ such that $M \not\models q$. This is usually stated as $D \cup \Sigma \models q$ if, and only if, $D \cup \Sigma \models_{\mathsf{fin}} q$ (where $\models_{\mathsf{fin}}$ stands for entailment under finite models), as the "only if" direction is always trivially true. And there are contexts, like in databases (Johnson and Klug 1984; Rosati 2006; Bárány *et al.* 2014), in which reasoning with respect to finite models is preferred.

In this paper, we focus on the Datalog$^{\pm}$ family, which has been introduced with the aim of "closing the gap between the Semantic Web and databases" (Calì *et al.* 2012) to provide the *Web of Data* with scalable formalisms that can benefit from existing database technologies. In fact, Datalog$^{\pm}$ generalizes two well-known subfamilies of description logics called $\mathscr{EL}$ and *DL-Lite*, which collect the basic tractable languages for OBQA in the context of the Semantic Web and databases. In particular, we consider ontologies where Σ is a set of *existential rules*, each of which is a first-order formula ρ of the form $\forall \mathbf{X} \forall \mathbf{Y} (\phi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z} p(\mathbf{X}, \mathbf{Z}))$, where the *body* $\phi(\mathbf{X}, \mathbf{Y})$ of ρ is a conjunction of atoms, and the *head* $p(\mathbf{X}, \mathbf{Z})$ of ρ is a single atom.

The main decidable Datalog$^{\pm}$ fragments rely on the following five syntactic properties: *weak-acyclicity* (Fagin *et al.* 2005), *guardedness* (Calì *et al.* 2013), *linearity* (Calì *et al.* 2012), *stickiness* (Calì *et al.* 2010), and *shyness* (Leone *et al.* 2012). And these properties underlie the basic classes called weakly-acyclic, guarded, linear, sticky, and shy, respectively. Several variants and combinations of these classes have been defined and studied too (Baget *et al.* 2010; Krötzsch and Rudolph 2011; Calì *et al.* 2012; Civili and Rosati 2012; Gottlob *et al.* 2013), as well as semantic properties subsuming the syntactic ones (Baget *et al.* 2009; Leone *et al.* 2012).

The five basic classes above are pairwise uncomparable, except for linear which is strictly contained in both guarded and shy, as depicted in Figure 1. Interestingly, both weakly-acyclic and shy strictly contain datalog—the well-known class with rules of the form $\forall \mathbf{X} \forall \mathbf{Y} (\phi(\mathbf{X}, \mathbf{Y}) \rightarrow p(\mathbf{X}))$, where existential quantification has been dropped. Moreover, sticky strictly contains joinless—the class collecting sets of rules where each body contains no repeated variable. The latter, introduced by Gogacz and Marcinkowski (2013) to prove that sticky is finitely controllable, plays a central role also in this paper. Finally, both linear and joinless strictly contain inclusion-dependencies—the well-known class of relational database dependencies collecting sets of rules with one single body atom and no repeated variable.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

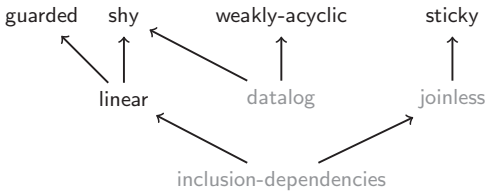


Fig. 1. Taxonomy of the basic Datalog$^{\pm}$ classes.

Under arbitrary models, OBQA can be reduced to the problem of answering q over a universal (or canonical) model U that can be homomorphically embedded into every other model (both finite and infinite) of $D \cup \Sigma$. Therefore, $D \cup \Sigma \models q$ if, and only if, $U \models q$. A way to compute a universal model is to employ the so called *chase* procedure. Starting from D, the chase "repairs" violations of rules by repeatedly adding new atoms—introducing fresh values, called *nulls*, whenever required by an existential variable—until a fixed point satisfying all rules is reached. In the classical setting, the chase is therefore sound and complete. But when *finite model reasoning* (namely, reasoning over finite models only, here denoted by $\models_{\mathsf{fin}}$) is required, then the chase is generally uncomplete, unless ontologies are finitely controllable. Hence, proving this property is of utmost importance, especially in those contexts where finite model reasoning is relevant.

Finite controllability of weakly-acyclic comes for free since every ontology here admits a finite universal model, computed by a variant of the chase procedure which goes under the name of *restricted chase* (Fagin *et al.* 2005). Conversely, the proof of this property for the subsequent three classes has been a very different matter. Complex, yet intriguing, constructions have been devised for linear (Rosati 2006; Bárány *et al.* 2014), guarded (Bárány *et al.* 2014), and more recently for sticky (Gogacz and Marcinkowski 2013). To complete the picture, we have addressed the same problem for shy and get the following positive result, which is the main contribution of the paper.

Theorem 1.1

Under shy ontologies, $D \cup \Sigma \models q$ if, and only if, $D \cup \Sigma \models_{\mathsf{fin}} q$.

For the proof, we design in Section 3 and exploit in Section 4 a general technique (our second contribution), called *canonical rewriting*, to facilitate the process of (dis)proving finite controllability of an arbitrary ontological fragment of existential rules. By exploiting this technique, we can immediately (re)confirm that linear is finitely controllable since inclusion-dependencies is. In addition, we prove (our third contribution) that sticky-join (Calì *et al.* 2012), generalizing both sticky and linear, is finitely controllable since sticky is. However, differently from linear and sticky-join, the canonical rewriting of a shy ontology—although it is simpler and still a shy ontology—does not immediately fall in any other known class. Therefore, to prove that shy is finitely controllable, we devise three technical tools on top of

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

the canonical rewriting from which we are able to exploit the fact that joinless is finitely controllable[1].

2 Ontology-based query answering

Basics. Let $\mathbf{C}$, $\mathbf{N}$, and $\mathbf{V}$ denote pairwise disjoint discrete sets of *constants*, *nulls* and *variables*, respectively. An element t of $\mathbf{T} = \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ is called *term*. An *atom* α is a labeled tuple $p(t_1, \ldots, t_m)$, where p is a predicate symbol, m is the *arity* of both p and α, and $t_1, \ldots, t_m$ are terms. An atom is *simple* if it contains no repeated term. We denote by $pred(\alpha)$ the predicate symbol p, and by $\alpha[i]$ the ith term t_i of the α. We also consider *propositional* atoms, which are simple atoms of arity 0 written without brackets. Given two sets A and B of atoms, a homomorphism from A to B is a mapping $h : \mathbf{T} \to \mathbf{T}$ such that $c \in \mathbf{C}$ implies $h(c) = c$, and also $p(t_1, \ldots, t_m) \in A$ implies $p(h(t_1), \ldots, h(t_m)) \in B$. As usual, we denote by $h(A) = \{p(h(t_1), \ldots, h(t_m)) : p(t_1, \ldots, t_m) \in A\} \subseteq B$. An *instance* I is a discrete set of atoms where each term is either a constant or a null.

Syntax. A *database* D is a finite null-free instance. An *(existential) rule* ρ is a first-order formula $\forall \mathbf{X} \forall \mathbf{Y}(\phi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z} p(\mathbf{X}, \mathbf{Z}))$, where $body(\rho) = \phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms, and $head(\rho) = p(\mathbf{X}, \mathbf{Z})$ is an atom. Constants may occur in ρ. If $\mathbf{Z} = \emptyset$, then ρ is datalog rule. An *ontology* Σ is a set of rules. For each rule ρ of Σ, we denote by $\mathbf{V}(\rho)$ the set of variables appearing in ρ, by $\mathbf{EV}(\rho)$ the set of all existential variables of ρ, and by $\mathbf{UV}(\rho)$ the set of all universal variables of ρ. A *union of Boolean conjunctive query*, UBCQ for short, q is a first-order expression of the form $\exists \mathbf{Y}_1 \psi_1(\mathbf{Y}_1) \vee \ldots \vee \exists \mathbf{Y}_k \psi_k(\mathbf{Y}_k)$, where each $\psi_j(\mathbf{Y}_j)$ is a conjunction of atoms. Constants may occur also in q. In case $k = 1$, then q is simply called BCQ.

Semantics. Consider a triple $\langle D, \Sigma, q \rangle$ as above. An instance I *satisfies* a rule $\rho \in \Sigma$, denoted by $I \models \rho$, if whenever there is a homomorphism h from $body(\rho)$ to I, then there is a homomorphism $h' \supseteq h|_{\mathbf{X}}$ from $\{head(\rho)\}$ to I. Moreover, I *satisfies* Σ, denoted by $I \models \Sigma$, if I satisfies each rule of Σ. The *models* of $D \cup \Sigma$, denoted by $mods(D, \Sigma)$, consist of the set $\{I : I \supseteq D \text{ and } I \models \Sigma\}$. An instance I *satisfies* q, written $I \models q$, if there is a homomorphism from some $\psi_j(\mathbf{Y}_j)$ to I. Also, q is *true* over $D \cup \Sigma$, written $D \cup \Sigma \models q$, if each model of $D \cup \Sigma$ satisfies q.

The chase. Consider a logical theory $\langle D, \Sigma \rangle$ as above. A rule ρ of Σ is *applicable* to an instance I if there is a homomorphism h from $body(\rho)$ to I that maps the existential variables of ρ to different nulls not occurring in I. If so, $\langle \rho, h \rangle(I) = I \cup h(head(\rho))$ defines a *chase step*. The *chase procedure* (Deutsch *et al.* 2008) of $D \cup \Sigma$ is any sequence $I_0 = D \subset I_1 \subset \ldots \subset I_m \subset \ldots$ of instances obtained by applying exhaustively the rules of Σ in a fair (e.g., breadth-first) fashion in such a way that, for each $i > 0$, $\langle \rho, h \rangle(I_{i-1}) = I_i$ defines a chase step for some ρ and h. We call $chase(D, \Sigma)$ the (possibly infinite) instance $\bigcup_{i \geqslant 0} I_i$. Importantly, different chase steps introduce different nulls. This variant of the chase is called *oblivious*, and defines a family of isomorphic instances, namely any two such instances are

[1] All complete proofs of the results presented in this paper can be found at the online Appendix or at the update version on arXiv (`https://arxiv.org/pdf/1707.01870.pdf`).

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

equal modulo renaming of nulls. Hence, without loss of generality, it is common practice to consider the oblivious chase as deterministic and its least fixpoint as unique. The *restricted* version of this procedure imposes a further condition on each chase step: $I \not\models h'(head(\rho))$, where $h' = h|_{\mathbf{UV}(\rho)}$. Differently from the oblivious one, it defines a family of homomorphically equivalent instances, each generically denoted by $rchase(D, \Sigma)$. It is well-known that $(r)chase(D, \Sigma)$ is a *universal* model of $D \cup \Sigma$, namely for each $M \in mods(D, \Sigma)$, there is a homomorphism from $chase(D, \Sigma)$ to M. Hence, given a UBCQ q, it holds that $(r)chase(D, \Sigma) \models q$ if, and only if, $D \cup \Sigma \models q$ (Fagin *et al.* 2005).

Finite controllability. The *finite models* of a theory $D \cup \Sigma$, denoted by $fmods(D, \Sigma)$, are the finite instances in $\{I \in mods(D, \Sigma) : |I| \in \mathbb{N}\}$. An ontological fragment $\mathscr{F}$ is *finitely controllable* if, for each database D, for each ontology Σ of $\mathscr{F}$, and for each UBCQ q, it holds that $D \cup \Sigma \not\models q$ implies that there exists a finite model M of $D \cup \Sigma$ such that $M \not\models q$. This is formally stated as $D \cup \Sigma \models q$ if and only if $D \cup \Sigma \models_{\mathsf{fin}} q$, or equivalently $chase(D, \Sigma) \models q$ if and only if $D \cup \Sigma \models_{\mathsf{fin}} q$.

2.1 *Datalog$^{\pm}$ fragments*

Fix a database D, an ontology Σ, and a chase step involving some pair $\langle \bar{\rho}, h \rangle$. To lighten the presentation, we assume that different rules of Σ share no variable. Also, for every m-ary predicate p and every $i \in \{1, \ldots, m\}$, the pair (p, i) is called *position* and denoted by $p[i]$. Finally, given a set A of atoms, a term t occurs in A at position $p[i]$ if there is $\alpha \in A$ s.t. $pred(\alpha) = p$ and $\alpha[i] = t$.

Local conditions. Σ belongs to (i) datalog whenever $\rho \in \Sigma$ implies $\mathbf{EV}(\rho) = \emptyset$; (ii) inclusion-dependencies whenever $\rho \in \Sigma$ implies that ρ contains only simple atoms and $|body(\rho)| = 1$; (iii) linear whenever $\rho \in \Sigma$ implies $|body(\rho)| = 1$; (iv) guarded whenever $\rho \in \Sigma$ implies that there is an atom of $body(\rho)$ containing all the variables of $\mathbf{UV}(\rho)$; (v) joinless whenever $\rho \in \Sigma$ implies that $head(\rho)$ is a simple atom and $body(\rho)$ contains no repeated variables.

Weak-acyclicity (Fagin et al. 2005). Informally, $\Sigma \in$ weakly-acyclic guarantees that if X occurs in $body(\bar{\rho})$ at position $p[i]$ and $h(X) \in \mathbf{N}$, then the number of distinct nulls occurring in $rchase(D, \Sigma)$ at position $p[i]$ are finitely many. Formally, the labeled graph $G(\Sigma)$ associated to Σ is defined as the pair $\langle N, A \rangle$, where (i) N collects all the positions $p[1], \ldots, p[m]$ for each m-ary predicate p occurring in Σ; (ii) $(p[i], r[j], \texttt{plain}) \in A$ if there is a rule $\rho \in \Sigma$ and a variable X of ρ such that X occurs in the body of ρ at position $p[i]$ and X occurs in the head of ρ at position $r[j]$; and (iii) $(p[i], r[j], \texttt{special}) \in A$ if there is a rule $\rho \in \Sigma$, a universal variable X occurring also in the head of ρ, and an existential variable Z of ρ such that X occurs in the body of ρ at position $p[i]$ and Z occurs in the head of ρ at position $r[j]$. Ontology Σ belongs to weakly-acyclic if $G(\Sigma)$ has no cycle going through an arc labeled as special.

Stickiness (Calì et al. 2012). Informally, $\Sigma \in$ sticky guarantees that if X occurs multiple times in $body(\bar{\rho})$, then X occurs in $head(\bar{\rho})$ and $h(X)$ belongs to every atom of $chase(D, \Sigma)$ that depends on $h(head(\bar{\rho}))$. Formally, a variable X of Σ is *marked* if (i) there is a rule $\rho \in \Sigma$ such that X occurs in $body(\rho)$ but not in $head(\rho)$; or

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

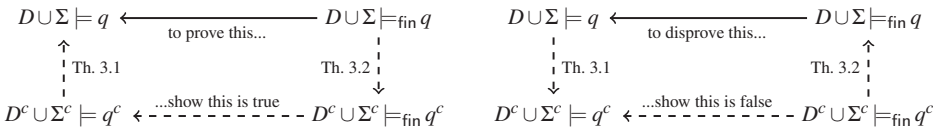


Fig. 2. Application of the canonical rewriting.

(ii) there are two rules $\rho, \rho' \in \Sigma$ such that a marked variable occurs in $body(\rho)$ at some position $p[i]$ and X occurs in $head(\rho')$ at position $p[i]$ too. Ontology Σ belongs to sticky if, for each $\rho \in \Sigma$, the following condition is satisfied: if X occurs multiple times in $body(\rho)$, then X is not marked. A more refined condition identifies interesting cases in which it is safe to allow rules containing some marked variable that occurs multiple times but in a single body atom only. This refinement gives rise to sticky-join, generalizing both sticky and linear.

Shyness (Leone et al. 2012). Informally, $\Sigma \in$ shy guarantees that (1) if X occurs in two different atoms of $body(\bar{\rho})$, then $h(X) \in \mathbf{C}$; and (2) if X and Y occur both in $head(\bar{\rho})$ and in two different atoms of $body(\bar{\rho})$, then $h(X) = h(Y)$ implies $h(X) \in \mathbf{C}$. Formally, consider an existential variable X of Σ. Position $p[i]$ is *invaded* by X if there is a rule ρ of Σ such that (i) X occurs in $head(\rho)$ at position $p[i]$, or (ii) some universal variable Y of ρ is *attacked* by X—namely Y occurs in $body(\rho)$ only at positions invaded by X—and it also occurs in $head(\rho)$ at position $p[i]$. A universal variable is *protected* if it is attacked by no existential variable. Ontology Σ belongs to shy if, for each $\rho \in \Sigma$, the following conditions are both satisfied: (1) if X occurs in two different atoms of $body(\rho)$, then X is protected; and (2) if X and Y occur both in $head(\rho)$ and in two different atoms of $body(\rho)$, then X and Y are not attacked by the same variable.

3 Canonical rewriting

In this section, we design a general technique to facilitate the process of (dis)proving finite controllability of an arbitrary ontological fragment of existential rules. More specifically, from a triple $\langle D, \Sigma, q \rangle$, we build the triple $\langle D^c, \Sigma^c, q^c \rangle$ enjoying the following properties: (1) D^c is propositional database; (2) Σ^c are constant-free rules containing only simple atoms; (3) q^c is a constant-free UBCQ with only simple atoms; (4) $chase(D^c, \Sigma^c)$ is a constant-free instance containing only simple atoms; and (5) there is a "semantic" correspondence between $mods(D, \Sigma)$ and $mods(D^c, \Sigma^c)$. By exploiting these properties, one can apply the technique shown in Figure 2.

3.1 Overview

Consider the database $D = \{person(tim), person(john), fatherOf(tim, john)\}$, and the ontology $\Sigma = \{person(X) \rightarrow \exists Y\, fatherOf(Y, X); fatherOf(X, Y) \rightarrow person(X)\}$. Let p, f, c_1, and c_2 be shorthands of *person*, *fatherOf*, *tim*, and *john*, respectively. Hence, $chase(D, \Sigma)$ is the instance $D \cup \{f(n_1, c_1), f(n_2, c_2)\} \cup \{p(n_i), f(n_{i+2}, n_i)\}_{i>0}$, where each n_i denotes a distinct null of $\mathbf{N}$. From D, we construct the propositional database

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

$D^c = \{p_{[c_1]}, p_{[c_2]}, f_{[c_1,c_2]}\}$ obtained by encoding in the predicates the tuples of D. Then, from Σ, we construct Σ^c collecting the following rules:

$$p_{[c_1]} \to \exists Y f_{[1,c_1]}(Y) \qquad f_{[c_1,c_1]} \to p_{[c_1]} \qquad f_{[c_1,1]}(Y) \to p_{[c_1]} \qquad f_{[1,1]}(X) \to p_{[1]}(X)$$
$$p_{[c_2]} \to \exists Y f_{[1,c_2]}(Y) \qquad f_{[c_1,c_2]} \to p_{[c_1]} \qquad f_{[c_2,1]}(Y) \to p_{[c_2]} \qquad f_{[1,2]}(X,Y) \to p_{[1]}(X)$$
$$p_{[1]}(X) \to \exists Y f_{[1,2]}(Y,X) \qquad f_{[c_2,c_1]} \to p_{[c_2]} \qquad f_{[1,c_1]}(X) \to p_{[1]}(X)$$
$$f_{[c_2,c_2]} \to p_{[c_2]} \qquad f_{[1,c_2]}(X) \to p_{[1]}(X)$$

The predicates here encode tuples of terms consisting of database constants (c_1 and c_2) and placeholders of nulls (1 and 2). Consider the first rule $\rho = p(X) \to \exists Y f(Y,X)$ applied by the chase over $D \cup \Sigma$, and $h = \{X \mapsto c_1, Y \mapsto n_1\}$ be its associated homomorphism. Hence, $h(body(\rho)) = p(c_1)$ and $h(head(\rho)) = f(n_1,c_1)$. Such an application is mimed by the "sister" rule $\rho^c = p_{[c_1]} \to \exists Y f_{[1,c_1]}(Y)$. By exploiting the same homomorphism, we obtain $h(body(\rho^c)) = p_{[c_1]}$ and also $h(head(\rho^c)) = f_{[1,c_1]}(n_1)$. Actually, the encoded tuple $[c_1]$ in $p_{[c_1]}$ says that the original twin atom $p(c_1)$ is unary and its unique term is exactly c_1. Moreover, the encoded tuple $[1,c_1]$ in $f_{[1,c_1]}(n_1)$ says that the original twin atom $f(n_1,c_1)$ is binary, that its first term is a null, and that its second term is exactly the constant c_1. Since from predicate $f_{[1,c_1]}$, we only know that the first term is a null, it must be unary to keep the specific null value. In the above construction, red rules are those applied by the chase on $D^c \cup \Sigma^c$. For example, rule $f_{[1,c_1]}(X) \to p_{[1]}(X)$ mimics $f(X,Y) \to p(X)$ when X is mapped to a null and Y to c_1; and rule $f_{[1,2]}(X,Y) \to p_{[1]}(X)$ mimics $f(X,Y) \to p(X)$ when X and Y are mapped to different nulls. Hence, $chase(D^c, \Sigma^c)$ is

$$D^c \cup \{p_{[1]}(n_i)\}_{i>0} \cup \{f_{[1,c_1]}(n_1), f_{[1,c_2]}(n_2)\} \cup \{f_{[1,2]}(n_{i+2}, n_i)\}_{i>0}.$$

As a result, the rewriting separates the interaction between the database constants propagated body-to-head via universal variables and the nulls introduced to satisfy existential variables. Also, since the predicates encode the "shapes" of the twin atoms—namely, $f_{[1,2]}(X,Y)$ means different nulls while $f_{[1,1]}(X)$ the same null—repeated variables are encoded too. By following the same approach, we can rewrite also the query. Consider for example the BCQ $q = \exists X p(X), f(X,c_1)$. Therefore, q^c is the UBCQ: $(p_{[c_1]}, f_{[c_1,c_1]}) \vee (p_{[c_2]}, f_{[c_2,c_1]}) \vee (\exists X\, p_{[1]}(X), f_{[1,c_1]}(X))$.

3.2 Formal construction and properties

Let us fix a triple $\langle D, \Sigma, q\rangle$ through the rest of this section. Consider an atom $\alpha = p(t_1,\ldots,t_m)$ with terms over $\mathbf{C} \cup \mathbf{V}$. The *canonical atom* of α is the atom $\alpha^c = p_{[\ell_1,\ldots,\ell_m]}(\tau_1,\ldots,\tau_\mu)$, where (a) $\ell_i = t_i$ if $t_i \in \mathbf{C}$; (b) $\ell_i = \ell_j$ if $t_i = t_j$; or (c) $\ell_i = 1 + \max(\{0\} \cup \{\ell_j \in \mathbb{N} : j < i\})$ if $t_i \in \mathbf{V}$ and $t_j \neq t_i\ \forall j < i$ and $\tau_i = V \in \mathbf{V}$, if there exists t_j such that $\ell_j = i$ and $t_j = V$. Moreover, given a set of atoms A, we define $A^c = \{\alpha^c : \alpha \in A\}$, and give a rule ρ, we define ρ^c as the rule so that $body(\rho^c) = body(\rho)^c$ and $head(\rho^c) = head(\rho)^c$. For instance, let $\alpha = p(c_1,X,c_2,X,Y,Z,Y)$ be an atom. Then, the canonical atom α^c of α is given by $p_{[c_1,1,c_2,1,2,3,2]}(X,Y,Z)$. Note that, by definition of τ_i, for $i = 1,\ldots,\mu$, we have that the arity $\mu \leqslant m$ of the canonical atom is equal to $\max(\{0\} \cup \{f(t_j) \in \mathbb{N} : j \leqslant m\})$.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

Definition 3.1 (*Safe and Canonical substitutions*)
A map $\varsigma : const(D \cup \Sigma) \cup \mathbf{V} \rightarrow const(D \cup \Sigma) \cup \mathbf{V}$ is called *canonical substitution* if $\varsigma(c) = c$ for each $c \in const(D \cup \Sigma)$. Moreover, we say that a canonical substitution ς is *safe* w.r.t. a rule $\rho \in \Sigma$ if $\varsigma(\mathbf{UV}(\rho)) \subseteq const(D \cup \Sigma) \cup \mathbf{UV}(\rho)$, and $\varsigma(V) = V$, for each $V \in \mathbf{EV}(\rho)$.

Intuitively, a safe substitution maps each existential variable to itself and no universal variable is mapped to an existental one. As usual, given a set of atoms A, we denote by $\varsigma(A) = \{p(\varsigma(t_1), \ldots, \varsigma(t_m)) : p(t_1, \ldots, t_m)\}$, and given a rule ρ, we denote by $\varsigma(\rho)$ the rule such that $body(\varsigma(\rho)) = \varsigma(body(\rho))$ and $head(\varsigma(\rho)) = \varsigma(head(\rho))$.

Example 3.1
Consider $D = \{r(c_1, c_3)\}$ and Σ consisting of the following rules: $\rho_1 = r(Y_1, Z_1), p(W_1, X_1, X_1, Y_1) \rightarrow \exists T_1 g(X_1, Y_1, T_1, X_1, Z_1)$ and $\rho_2 = s(X_2), t(Y_2) \rightarrow r(X_2, Y_2)$. For instance, $\varsigma_1 = \{c_1 \mapsto c_1, c_3 \mapsto c_3, Y_1 \mapsto X_1,\ Z_1 \mapsto c_3,\ W_1 \mapsto Y_1,\ X_1 \mapsto X_1,\ T_1 \mapsto T_1\}$ and $\varsigma_1' = \{c_1 \mapsto c_1, c_3 \mapsto c_3, Y_1 \mapsto c_1,\ Z_1 \mapsto X_1,\ W_1 \mapsto c_1,\ X_1 \mapsto Y_1,\ T_1 \mapsto T_1\}$ are safe substitutions w.r.t. ρ_1. Indeed, $const(D \cup \Sigma) = \{c_1, c_3\}$, $\mathbf{UV}(\rho_1) = \{W_1, X_1, Y_1, Z_1\}$, $\mathbf{EV}(\rho_1) = \{T_1\}$, the existential variable T_1 is mapped to itself, and no other variable is mapped to an existential one. Moreover, $\varsigma_1(\rho_1) = r(X_1, c_3), p(Y_1, X_1, X_1, X_1) \rightarrow \exists T_1 g(X_1, X_1, T_1, X_1, c_3)$ and $\varsigma_1'(\rho_1) = r(c_1, X_1), p(c_1, Y_1, Y_1, c_1) \rightarrow \exists T_1 g(Y_1, c_1, T_1, Y_1, X_1)$. ◁

We denote by $\mathscr{CS}$ the set of all canonical substitutions and by $ß(\rho) \subseteq \mathscr{CS}$ the set of all safe substitutions w.r.t. ρ. Given a set of atoms A [resp. a rule ρ] and a canonical substitution [resp. safe substitution] ς, we say that $\varsigma(A)^c$ [resp. $\varsigma(\rho)^c$] is the *canonical set of atoms* w.r.t A [resp. *canonical rule w.r.t.* ρ] and ς. Observe that two different canonical substitutions could produce two isomorphic canonical set of atoms. For instance, let $A = \{p(X, Y)\}$, and consider $\varsigma = \{X \mapsto X,\ Y \mapsto Y\}$ and $\varsigma' = \{X \mapsto Y,\ Y \mapsto X\}$. Then, $\varsigma(A)^c = \{p_{[1,2]}(X, Y)\}$ and $\varsigma'(A)^c = \{p_{[1,2]}(Y, X)\}$ are isomorphic set of atoms. Therefore, to avoid redundancies, we denote by $\mathscr{CS}^*$ [resp. $ß^*(\rho)$] any arbitrary maximal subset of $\mathscr{CS}$ [resp. of $ß(\rho)$] producing canonical set of atoms [resp. canonical rules] containing no two isomorphic elements.

We denote by Σ^c the set of all canonical rules $\{\varsigma(\rho)^c : \rho \in \Sigma \text{ and } \varsigma \in ß^*(\rho)\}$, and we call it the *canonical rewriting* of Σ. Also, given a UBCQ q of the form $\exists \mathbf{Y}_1 \psi_1(\mathbf{Y}_1) \vee \ldots \vee \exists \mathbf{Y}_k \psi_k(\mathbf{Y}_k)$, we denote by q^c the disjunction $\bigvee_{\varsigma_1 \in \mathscr{CS}^*} \varsigma_1(\psi_1(\mathbf{Y}_1))^c \vee \ldots \vee \bigvee_{\varsigma_k \in \mathscr{CS}^*} \varsigma_k(\psi_k(\mathbf{Y}_k))^c$ and we call it the *canonical rewriting* of q. Finally, we call D^c the *canonical rewriting* of D.

Proposition 3.1
The triple $\langle D^c, \Sigma^c, q^c \rangle$ can be constructed from $\langle D, \Sigma, q \rangle$ in polynomial time (in data complexity).

Example 3.2
Consider the ontology Σ with the safe substitutions ς_1 and ς_1' w.r.t. ρ_1 of the Example 3.1. Therefore, we obtain the canonical rules: $\varsigma_1(\rho_1)^c = r_{[1,c_3]}(X_1), p_{[1,2,2,2]}(Y_1, X_1) \rightarrow \exists T_1 g_{[1,1,2,1,c_3]}(X_1, T_1)$ and $\varsigma_1'(\rho_1)^c = r_{[c_1,1]}(X_1), p_{[c_1,1,1,c_1]}(Y_1) \rightarrow \exists T_1 g_{[1,c_1,2,1,3]}(Y_1, T_1, X_1)$. Moreover, let ς_2 and ς_2' be the safe substitutions containing $\{X_2 \mapsto X_2, Y_2 \mapsto$

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

$X_2\}$ and $\{X_2 \mapsto c_1, Y_2 \mapsto c_3\}$ w.r.t. ρ_2, respectively. Hence, we have $\varsigma_2(\rho_2)^c = s_{[1]}(X_2), t_{[1]}(X_2) \rightarrow r_{[1,1]}(X_2)$ and $\varsigma_2'(\rho_2)^c = s_{[c_1]}, t_{[c_3]} \rightarrow r_{[c_1,c_3]}$. Therefore, $\varsigma_1(\rho_1)^c$, $\varsigma_1'(\rho_1)^c$, $\varsigma_2(\rho_2)^c$ and $\varsigma_2'(\rho_2)^c$ are (some of the) rules of Σ^c. ◁

We consider a function $\mathfrak{R}$ from the set of atoms of $D^c \cup \Sigma^c$ to the set of atom of $D \cup \Sigma$. For each atom $\alpha = a_{[s_1,\ldots,s_m]}(\sigma_1, \ldots, \sigma_\mu)$, we build an atom $\mathfrak{R}(\alpha) = a(t_1, \ldots, t_m)$ such that (a) $t_i = s_i$ if $s_i \in \mathbf{C}$; (b) $t_i = \sigma_i$ if $s_i = k$ and $s_j \neq k$, for each $j < i$; or (c) $t_i = \sigma_j$ if $s_i = s_j$, for some $j < i$.

For instance, let $\alpha = p_{[1,c_1,2,1,c_2,1,2]}(X, Y)$ be an atom of the logical theory $D^c \cup \Sigma^c$. Then, $\mathfrak{R}(\alpha) = p(X, c_1, Y, X, c_2, X, Y)$. We call $\mathfrak{R}$ the *unpacking function*. Given a set of atoms A of $D^c \cup \Sigma^c$, we denote by $\mathfrak{R}(A) = \{\mathfrak{R}(\alpha) : \alpha \in A\}$ the corresponding set of atoms of $D \cup \Sigma$. If I is an instance, we call $\mathfrak{R}(I)$ the *unpacked instance* of I. Given a rule ρ^c in Σ^c, we denote by $\mathfrak{R}(\rho^c)$ the rule obtained applying $\mathfrak{R}$ to each atom in ρ^c, i.e., $\mathfrak{R}(\rho^c) : \mathfrak{R}(body(\rho^c)) \rightarrow \mathfrak{R}(head(\rho^c))$, and we call it the *unpacked rule* of ρ^c. Similarly, we denote by $\mathfrak{R}(q^c)$ the query obtained applying $\mathfrak{R}$ to the atoms of the UBCQ q^c, and we call it the *unpacked query* of q^c. Informally, the unpacking function acts as the inverse operator to the canonical rewriting. Moreover, it enjoys an interesting and useful property: the chase of a logical theory coincides with the unpacking of the chase constructed from of the same theory given in canonical form:

Proposition 3.2
Consider a set Σ of existential rules. For each database D and for each UBCQ q, it holds that $\mathfrak{R}(chase(D^c, \Sigma^c)) = chase(D, \Sigma)$ and $\mathfrak{R}(q^c) \equiv q$.

By exploiting the above proposition, we can now prove that a UBCQ q is satisfied by all models of a theory $D \cup \Sigma$ if, and only if, each model of the canonical rewriting of the theory $D^c \cup \Sigma^c$ satisfies the canonical rewriting of the UBCQ q^c.

Theorem 3.1
$D \cup \Sigma \models q$ if, and only if, $D^c \cup \Sigma^c \models q^c$.

Note that, if Σ is a constant-free ontology, then, for each model M^c of $D^c \cup \Sigma^c$, $\mathfrak{R}(M^c)$ is a model of $D \cup \Sigma$. The request for a constant-free ontology is needed. Indeed, for instance, let $\Sigma = \{p(a) \rightarrow r(a);\ r(x) \rightarrow p(x)\}$. So that, $\Sigma^c = \{p_{[a]} \rightarrow r_{[a]};\ r_{[a]} \rightarrow p_{[a]};\ r_{[1]}(V_1) \rightarrow p_{[1]}(V_1)\}$. Therefore, $M^c = \{p_{[1]}(a)\}$ is a model of Σ^c, but $\mathfrak{R}(M^c) = \{p(a)\}$ is not a model of Σ, as it does not satisfy the first rule. However, we can overcome this problem considering the following class of models. Given a model $M^c \in mods(D^c, \Sigma^c)$, we say that N^c is a *smooth* instance of M^c if there exists a bijective map $f : terms(M^c) \rightarrow terms(N^c)$ such that $f(n) = n$ for each null $n \in terms(M^c)$; $f(c) = n_c$ for each constant $c \in terms(M^c)$, where n_c is a fresh null; and $f(M^c) = N^c$. Note that a smooth instance of a model M^c is also a model of $D^c \cup \Sigma^c$ and it is also constant free.

Proposition 3.3
If $M^c \in mods(D^c, \Sigma^c)$, then $\mathfrak{R}(N^c) \in mods(D, \Sigma)$, for each smooth model N^c of M^c.

By exploiting the above proposition, we can now prove that a UBCQ q is satisfied by all finite models of a theory $D \cup \Sigma$ if, and only if, each finite model of the

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

canonical rewriting of the theory $D^c \cup \Sigma^c$ satisfies the canonical rewriting of the UBCQ q^c.

Theorem 3.2

$D \cup \Sigma \models_{\mathsf{fin}} q$ if, and only if, $D^c \cup \Sigma^c \models_{\mathsf{fin}} q^c$.

Proof

Assume that $D \cup \Sigma \models_{\mathsf{fin}} q$. Then, for each finite model M of $D \cup \Sigma$, there exists a homomorphism h from at least one disjunct of q, say $\psi_j(\mathbf{Y}_j)$ to M. Now, let M^c be a finite model of $D^c \cup \Sigma^c$. By Proposition 3.3, there exist a (finite) smooth model $N^c \in \mathit{mods}(D^c, \Sigma^c)$ of M^c and a bijective map f from $\mathit{terms}(M^c)$ to $\mathit{terms}(N^c)$ such that $f(n) = n$ for each null $n \in \mathit{terms}(M^c)$; $f(c) = n_c$ for each constant $c \in \mathit{terms}(M^c)$, where n_c is a fresh null; $f(M^c) = N^c$, and $\mathfrak{R}(N^c) \in \mathit{mods}(D, \Sigma)$. Hence, by assumption, there exists a homomorphism h from some $\psi_j(\mathbf{Y}_j)$ to $\mathfrak{R}(N^c)$. Let $A = h(\psi_j(\mathbf{Y}_j)) \subseteq \mathfrak{R}(N^c)$. Then, for each atom $\alpha \in A$, we can choose an arbitrary atom $\beta \in N^c$ such that $\mathfrak{R}(\beta) = \alpha$. Let B such a subset of N^c. Therefore, by construction, there exists a BCQ in q^c isomorphic to B. In particular, there exists a homomorphism h from q^c to N^c. In conclusion, $f^{-1} \circ h$ is a homomorphism from q^c to M^c. Indeed, $f^{-1} \circ h$ is a map from $\mathit{terms}(q^c)$ to $\mathit{terms}(M^c)$ such that $f^{-1}(h(q^c)) \subseteq f^{-1}(N^c) = M^c$. Now, assume that $D^c \cup \Sigma^c \models_{\mathsf{fin}} q^c$. Let M be a finite model of $D \cup \Sigma$. By definition of canonical rules, it can be easily proved that there exists a finite model $M^c \in \mathit{fmods}(D^c, \Sigma^c)$ such that $\mathfrak{R}(M^c) = M$. Hence, let h be a homomorphism from q^c to M^c. So that, $h(\varsigma_j(\psi_j(\mathbf{Y}_j))^c) \subseteq M^c$, for some disjunct $\varsigma_j(\psi_j(\mathbf{Y}_j))^c$ of q^c. Therefore, by applying the unpacked function, we have that $\mathfrak{R}(h(\varsigma_j(\psi_j(\mathbf{Y}_j))^c)) = h(\mathfrak{R}(\varsigma_j(\psi_j(\mathbf{Y}_j))^c)) = h(\varsigma_j(\psi_j(\mathbf{Y}_j))) \subseteq \mathfrak{R}(M^c) = M$. Hence, h is also a homomorphism from q to M. □

3.3 Immediate consequences

By exploiting the properties of the canonical rewriting, one can reprove that linear is finitely controllable, and prove (for the first time) that also sticky-join enjoys this property. In fact, given a linear or sticky-join ontology Σ, its canonical rewriting Σ^c belongs to inclusion-dependencies or sticky, respectively. In the former case, it suffices to observe that any variable occurring multiple times in some atom α, by definition, occurs exactly once in its associated canonical atom α^c. In the latter case, additionally, consider a variable X violating the sticky property since it is marked and it occurs multiple times in the body of some rule ρ. By hypothesis, X may occur in exactly one atom of $\mathit{body}(\rho)$. However, even if marked, X now occurs exactly once in its canonical atom and it cannot violate the sticky property anymore. The following result follows.

Theorem 3.3

Under sticky-join ontologies, $D \cup \Sigma \models q$ if, and only if, $D \cup \Sigma \models_{\mathsf{fin}} q$.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

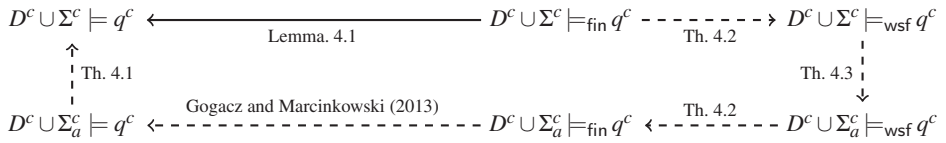


Fig. 3. Chain of implications for the proof of Lemma 4.1.

4 Finite controllability of shy ontologies

We open this section by observing that, differently from linear and sticky-join, the canonical rewriting of a shy ontology—although it is still a shy ontology—does not fall in any other known class. To prove that shy is finitely controllable, we therefore devise three technical tools on top of the canonical rewriting defined in Section 3. These tools allow us to show that $D^c \cup \Sigma^c \models_{\mathsf{fin}} q^c$ if, and only if, $D^c \cup \Sigma^c \models q^c$ (Lemma 4.1). To this end, let us fix a triple $\langle D, \Sigma, q \rangle$, and the associated one $\langle D^c, \Sigma^c, q^c \rangle$ in canonical form. Our tools are as follows:

Active and harmless rules. Whenever Σ is shy, we can partition Σ^c in two sets, denoted by Σ_a^c and Σ_h^c—collecting *active* and *harmless* rules, respectively—enjoying the following properties: (1) Σ_h^c are the rules of Σ^c with at least a variable occurring in more than one body atom; (2) $\Sigma_a^c = \Sigma^c \setminus \Sigma_h^c$ is a joinless (and still shy) ontology; and (3) $chase(D^c, \Sigma^c) = chase(D^c, \Sigma_a^c)$.

Well-supported finite models. Inspired by well-supported interpretations of general logic programs (Fages 1991), we define *well-supported* finite models of $\langle D, \Sigma \rangle$, denoted by $wsfmods(D, \Sigma)$, which enjoy the following properties: (1) for each $M \in wsfmods(D, \Sigma)$, there exists an *ordering* $(\alpha_1, \ldots, \alpha_m)$ of its atoms such that, for each α_j of M, either α_j belongs to D, or there exists a rule $\rho \in \Sigma$ and a homomorphism from the atoms of ρ to $\{\alpha_1, \ldots, \alpha_j\}$ that maps $body(\rho)$ to $\{\alpha_1, \ldots, \alpha_{j-1}\}$ and $head(\rho)$ to $\{\alpha_j\}$; (2) for each $M \in fmods(D, \Sigma)$, there exists a well-supported finite model $M' \subseteq M$; and (3) each minimal finite model of $D \cup \Sigma$ is a well-supported finite model.

Propagation ordering. Since $mods(D^c, \Sigma^c) \subseteq mods(D^c, \Sigma_a^c)$, in general, it is definitely possible that a model M of $D^c \cup \Sigma_a^c$ is not a model of $D^c \cup \Sigma^c$. In case Σ is shy and M is a well-supported finite models of $D^c \cup \Sigma_a^c$, by exploiting an arbitrary ordering of M, we show how to rename and propagate some of the terms of M to construct an instance M' enjoying the following property: (1) $M' \in wsfmods(D^c, \Sigma^c)$; and (2) there exists a homomorphism from M' to M.

With these tools in place, we can now apply the technique shown in Figure 3, where we use the symbol $\models_{\mathsf{wsf}}$ to refer the satisfiability of the query under well-supported finite models only.

4.1 Active and harmless rules

As said, and next stated, the canonical rewriting of a shy ontology is again a shy ontology.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

Proposition 4.1
If Σ is shy, then Σ^c is.

The goal of this section is therefore to identify a suitable subset of Σ^c that falls in some known finitely controllable class, and that roughly "behaves" as Σ^c under both finite and arbitrary models. The idea is to collect in Σ^c_h the rules of Σ^c with at least a variable occurring in more than one body atom, and to define $\Sigma^c_a = \Sigma^c \setminus \Sigma^c_h$. In other words, Σ^c_a is exactly the maximal subset of Σ^c that belongs to joinless. Let us now provide some insights regarding this way of partitioning Σ^c. From the database $D = \{p(c)\}$ and the shy ontology $\Sigma = \{p(X) \rightarrow \exists Y\ f(Y,X);\ f(X,Y), p(X) \rightarrow p(Y)\}$, we first construct $D^c = \{p_{[c]}\}$ and Σ^c as the following set of rules:

$$
\begin{array}{ccc}
f_{[1,c]}(X), p_{[1]}(X) \rightarrow p_{[c]} & f_{[c,c]}, p_{[c]} \rightarrow p_{[c]} & p_{[c]} \rightarrow \exists Y\, f_{[1,c]}(Y) \\
f_{[1,1]}(X), p_{[1]}(X) \rightarrow p_{[1]}(X) & f_{[c,1]}(Y), p_{[c]} \rightarrow p_{[1]}(Y) & p_{[1]}(X) \rightarrow \exists Y\, f_{[1,2]}(Y,X) \\
f_{[1,2]}(X,Y), p_{[1]}(X) \rightarrow p_{[1]}(Y) & &
\end{array}
$$

Again, red rules are those applied by the chase on $D^c \cup \Sigma^c$. Now we observe that there is no way to trigger the rules in the first column: although the chase does produce an atom $f(t,n_i)$ for some term t and null n_i, it never produces any atom $p(n_i)$. This fact is detected by the syntactic conditions underlying shy (marking X in $f(X,Y), p(X) \rightarrow p(Y)$ as "protected"), which guarantee that X may be mapped by the chase to constants only. Hence, since by definition Σ^c_a consists of the joinless rules in the last two columns, it holds that $chase(D^c, \Sigma^c) = chase(D^c, \Sigma^c_a)$.

The reason underlying the fact that the chase never applies rules of Σ^c_h will be exploited in Section 4.3 to prove Theorem 4.3 (see Fig. 3, right-hand side), namely that Σ^c_a roughly "behaves" as Σ^c under finite (well-supported) models. Conversely, to show Theorem 4.1 (see Fig. 3, left-hand side), it suffices to observe the more general property that $\Sigma^c_a \subseteq \Sigma^c$, which immediately implies $mods(D^c, \Sigma^c) \subseteq mods(D^c, \Sigma^c_a)$. And the next result follows.

Theorem 4.1
If $D^c \cup \Sigma^c_a \models q^c$, then $D^c \cup \Sigma^c \models q^c$.

4.2 Well-supported finite models

We start by defining the notion of well-supported finite instances, which is inspired by the related notion of well-supported interpretations for general logic programs (Fages 1991).

Let D be a database, and Σ be an ontology. A finite instance I is called *well-supported* w.r.t. the theory $D \cup \Sigma$ if there is an ordering $(\alpha_1, \ldots, \alpha_m)$ of its atoms such that, for each $j \in \{1, \ldots, m\}$, at least one of the following conditions is satisfied: (1) α_j is a database atom of D; and (2) there exists a rule ρ of Σ and a homomorphism h from $atoms(\rho)$ to $\{\alpha_1, \ldots, \alpha_j\}$ such that $h(head(\rho)) = \{\alpha_j\}$ and $h(body(\rho)) \subseteq \{\alpha_1, \ldots, \alpha_{j-1}\}$. In both cases, we will say that α_j is a *well-supported atom* w.r.t. $(\alpha_1, \ldots, \alpha_m)$; while in the latter case, we will also say that ρ is a *well-supporting rule* for α_j w.r.t. $(\alpha_1, \ldots, \alpha_m)$. Such an ordering will be called a *well-supported ordering* of I.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

We denote by $wsfmods(D,\Sigma) \subseteq fmods(D,\Sigma)$ the set of all well-supported finite models of $D \cup \Sigma$. Moreover, if a UBCQ q is satisfied by each model of $wsfmods(D,\Sigma)$, we write $D \cup \Sigma \models_{\mathsf{wsf}} q$. Interestingly, each finite model of $D \cup \Sigma$ contains a well-supported finite model of the theory.

Proposition 4.2
For each $M \in fmods(D,\Sigma)$, there exists $M' \subseteq M$ such that $M' \in wsfmods(D,\Sigma)$. In particular, each minimal finite model of $D \cup \Sigma$ is a well-supported finite model.

Although each finite model of an ontological theory contains a well-supported finite model of the theory, the reverse inclusion does not hold. Consider for example the ontology Σ of Section 3.1, and the model $M = D \cup \{f(c_1,c_1), f(c_2,c_1)\}$. Since $(p(c_1), p(c_2), f(c_1,c_2), f(c_1,c_1), f(c_2,c_1))$ is a well-supported ordering of M, then M is well-supported. However, $M \setminus \{f(c_2,c_1)\}$ is a model of $D \cup \Sigma$. Therefore, M is not a minimal one. Using Proposition 4.2, we can now prove that if a UBCQ q can be satisfied by each well-supported finite model of a theory, then it can be satisfied by each finite model of the theory.

Theorem 4.2
$D \cup \Sigma \models_{\mathsf{wsf}} q$ if, and only if, $D \cup \Sigma \models_{\mathsf{fin}} q$.

Proof
Clearly, by subset inclusion, if each finite model of $D \cup \Sigma$ satisfies the query q, then each well-supported finite model of $D \cup \Sigma$ satisfies q. Moreover, as each finite minimal model is a well-supported finite model (Proposition 4.2), then for each finite model M' of $D \cup \Sigma$, we can find a well-supported finite model, that is minimal, M of $D \cup \Sigma$, such that $M \subseteq M'$, and, in particular, there exists a homomorphism h (i.e., the identity homomorphism) such that $h(M) \subseteq M'$. □

4.3 Propagation ordering

Let us start with the preliminary notions of existentially well-supported atom and propagated term. Let I be a well-supported finite instance, and $(\alpha_1,\ldots,\alpha_m)$ be a well-supported ordering of I. An atom α of $I \setminus D$ is said *existentially well-supported* w.r.t. the ordering $(\alpha_1,\ldots,\alpha_m)$ if, for each well-supporting rule ρ for α w.r.t. $(\alpha_1,\ldots,\alpha_m)$, it holds that $\mathbf{EV}(\rho) \neq \emptyset$. Moreover, let $\alpha_j[k] = t$, for some position k, then t is said *propagated* from an atom α_i in position l, whenever $i < j$, $\alpha_i[l] = t$, and there exists a well-supporting rule ρ for α_j and a homomorphism h such that $\alpha_i \in h(body(\rho))$. Consider again ontology Σ of Section 3.1, and the well-supported finite model M considered after Proposition 4.2. For instance, the atom $f(c_1,c_1)$ is existentially well-supported. Indeed, the unique way to well-support the atom comes from the first rule of Σ, that is an existential rule. We are now ready to define the notion of propagation ordering.

Definition 4.1 (*Propagation ordering*)
Let D be a database, Σ be a joinless ontology, $M \in wsfmods(D,\Sigma)$, and $(\alpha_1,\ldots,\alpha_m)$ be a well-supported ordering of M. For each $\alpha_j \in M$, we build a new atom $\langle\alpha_j\rangle$

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

as follows. Let $t = \alpha_j[k]$. We have the following: (1) If α_j is an existentially well-supported atom and k is an existential position, then $\langle\alpha_j\rangle[k] = \langle t, j, k\rangle$, where $\langle t, j, k\rangle$ is called a *starting point* of t; (2) If t is a propagated term from some atom α_i in position l, then $\langle\alpha_j\rangle[k] = \langle\alpha_i\rangle[l]$; and (3) $\langle\alpha_j\rangle[k] = \alpha_j[k]$, otherwise. We call $(\langle\alpha_1\rangle, \ldots, \langle\alpha_m\rangle)$ a *propagation ordering* of the well-supported ordering $(\alpha_1, \ldots, \alpha_m)$.

Note that the same term could have several starting points. This propagation ordering will be useful to remember a starting point of that particular term and its propagations in other atoms.

Example 4.1
Consider the following joinless ontology $\Sigma = \{s(X_1) \rightarrow \exists Y_1 p(X_1, Y_1);\ s(X_2) \rightarrow \exists Y_2 u(Y_2, X_2);\ p(X_3, Y_3), u(W_3, Z_3) \rightarrow r(Y_3, Z_3);\ p(X_4, Y_4) \rightarrow t(Y_4)\}$, and the database $D = \{s(c_1)\}$. As an example, $M = \{s(c_1), t(c_2), t(n_1), p(c_1, c_2), p(c_1, n_1), r(c_2, c_1), r(n_1, c_1), u(c_2, c_1), u(n_1, c_1)\}$ is a well-supported finite model of $D \cup \Sigma$. Indeed, for instance, $(s(c_1), p(c_1, c_2),\ p(c_1, n_1),\ u(c_2, c_1),\ t(c_2),\ u(n_1, c_1),\ r(n_1, c_1),\ t(n_1), r(c_2, c_1))$ is a well-supported ordering of M. The existentially well-supported atoms are $p(c_1, c_2)$, $p(c_1, n_1)$, $u(c_2, c_1)$ and $u(n_1, c_1)$. More specifically, $p(c_1, c_2)$ has the term c_2 in the existential position 2, then $\langle p(c_1, c_2)\rangle = p(c_1, \langle c_2, 2, 2\rangle)$, as $p(c_1, c_2)$ is the second atom of the well-supported ordering considered; $p(c_1, n_1)$ has the term n_1 in the existential position 2, then $\langle p(c_1, n_1)\rangle = p(c_1, \langle n_1, 3, 2\rangle)$; $u(c_2, c_1)$ has the term c_2 in the existential position 1, then $\langle u(c_2, c_1)\rangle = u(\langle c_2, 4, 1\rangle, c_1)$; $u(n_1, c_1)$ has the term n_1 in the existential position 1, then $\langle u(n_1, c_1)\rangle = u(\langle n_1, 6, 1\rangle, c_1)$. On the other hand, the term c_2 is propagated in the atom $t(c_2)$ in the first (and unique) position. It comes from atom $p(c_1, c_2)$, and we know that the starting point of c_2 is $\langle c_2, 2, 2\rangle$. Therefore, $\langle t(c_2)\rangle = t(\langle c_2, 2, 2\rangle)$. Moreover, in a similarly way, we obtain that $\langle t(n_1)\rangle = t(\langle n_1, 3, 2\rangle)$. Finally, the term n_1 is propagated in the atom $r(n_1, c_1)$ in the first position, and it comes from atom $p(c_1, n_1)$; whereas the term c_1 is propagated in the atom $r(n_1, c_1)$ in the second position, and it comes from atom $u(n_1, c_1)$. Therefore, $\langle r(n_1, c_1)\rangle = r(\langle n_1, 3, 2\rangle, c_1)$. $\lhd$

With our technical tools in place, we are now able to prove the following technical result.

Theorem 4.3
For each $\Sigma \in$ shy, if $D^c \cup \Sigma^c \models_{\mathsf{wsf}} q^c$, then $D^c \cup \Sigma^c_a \models_{\mathsf{wsf}} q^c$.

Proof intuition
Consider an arbitrary model $M \in wsfmods(D^c, \Sigma^c_a)$. It suffices to prove that there exists $M' \in wsfmods(D^c, \Sigma^c)$ and a homomorphism h' s.t. $h'(M') \subseteq M$. Indeed, by hypothesis, there exists a homomorphism h s.t. $h(q) \subseteq M'$, and so $(h' \circ h)(q) \subseteq M$.

The difficulty here is that M could not be a model of $D^c \cup \Sigma^c$. Consider the database $D = \{s(c)\}$ and the shy ontology $\Sigma = \{s(X) \rightarrow \exists Y\, p(Y);\ s(X) \rightarrow \exists Y\, r(Y);\ p(X), r(X) \rightarrow g(X)\}$. The canonical rewriting is $D^c = \{s_{[c]}\}$ and Σ^c as follows:

$$\begin{array}{ccc} s_{[c]} \rightarrow \exists Y\, p_{[1]}(Y) & s_{[c]} \rightarrow \exists Y\, r_{[1]}(Y) & p_{[c]}, r_{[c]} \rightarrow g_{[c]} \\ s_{[1]}(X) \rightarrow \exists Y\, p_{[1]}(Y) & s_{[1]}(X) \rightarrow \exists Y\, r_{[1]}(Y) & p_{[1]}(X), r_{[1]}(X) \rightarrow g_{[1]}(X) \end{array}$$

One can verify that $M = \{s_{[c]}, p_{[1]}(n_1), r_{[1]}(n_1)\}$ is a (minimal) well-supported finite model of $D^c \cup \Sigma^c_a$ since, by Proposition 4.1, Σ^c is shy, and since Σ^c_a is obtained from

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

Σ^c by discarding the last harmless rule. However, M is not a model of $D^c \cup \Sigma^c$ because the last rule is not satisfied.

The idea is to show how to construct from M a model $M' \in wsfmods(D^c, \Sigma^c)$ that can be homomorphically mapped to M. Intuitively, we identify the *starting points* in which existential variables of Σ_a^c have been satisfied and rename the introduced terms using a propagation ordering.

In the example above, consider the well-supported ordering $(s_{[c]}, p_{[1]}(n_1), r_{[1]}(n_1))$ of M, replace n_1 in $p_{[1]}(n_1)$ by $\langle n_1, 2, 2\rangle$ (null n_1 introduced in the second atom in the second position), and replace n_1 in $r_{[1]}(n_1)$ by $\langle n_1, 3, 2\rangle$ (null n_1 introduced in the third atom in the second position). Then, since M is well-supported, we propagate (if needed) these new terms according to the supporting ordering. In our case, $M' = \{s_{[c]}, p_{[1]}(\langle n_1, 2, 2\rangle), r_{[1]}(\langle n_1, 3, 2\rangle)\}$ is now a finite model of $D^c \cup \Sigma^c$ that can be mapped to M. □

4.4 The main result

Lemma 4.1
Under shy ontologies, $D^c \cup \Sigma^c \models q^c$ if, and only if, $D^c \cup \Sigma^c \models_{\mathsf{fin}} q^c$.

Proof
Clearly, the "only if" implication is straightforward. Hence, given a shy ontology Σ, we have to prove that $D^c \cup \Sigma^c \models q^c$, whenever $D^c \cup \Sigma^c \models_{\mathsf{fin}} q^c$, for each database D and UBCQ q. Suppose that $D^c \cup \Sigma^c \models_{\mathsf{fin}} q^c$, i.e., the query q^c is satisfied by each finite model of $D^c \cup \Sigma^c$. Thus, by Theorem 4.2, holds that $D^c \cup \Sigma^c \models_{\mathsf{wsf}} q^c$, that is, the canonical rewriting of the query q is satisfied by each well-supported finite model of the logical theory $D^c \cup \Sigma^c$. Then, by Theorem 4.3, holds that $D^c \cup \Sigma_a^c \models_{\mathsf{wsf}} q^c$, that is, the canonical rewriting of the query q is satisfied by each well-supported finite model of the joinless logical theory $D^c \cup \Sigma_a^c$. Moreover, again, by Theorem 4.2, we obtain that $D^c \cup \Sigma_a^c \models_{\mathsf{fin}} q^c$, that is, q^c is satisfied also by every finite model of the previous theory. Now, as Σ_a^c is a joinless ontology, by the finite controllability of joinless ontologies proved by Gogacz and Marcinkowski (2013), holds that $D^c \cup \Sigma_a^c \models q^c$. Finally, by Theorem 4.1, we have that $D^c \cup \Sigma^c \models q^c$, i.e., the query q^c is satisfied by each model (finite or infinite) of $D^c \cup \Sigma^c$. □

Summing up, Theorem 1.1 follows by combining Lemma 4.1 with the properties of the canonical rewriting proved in Section 3.

5 Related work

To complete the related works started with the Introduction, we recall that the notion of finite controllability was formalized for the first time by Rosati (2006) while he was working on a question that had been left open two decades before by Johnson and Klug (1984) about containment of conjunctive queries in case of both arbitrary and finite databases. Basically, using our terminology, they proved that ontologies mixing both inclusion dependencies and functional dependencies are

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

not finitely controllable, by leaving open the case where ontologies contain inclusion dependencies only. Rosati then answered positively this question.

The semantic equivalence of fundamental reasoning tasks under finite and infinite models is not at all a prerogative of the database community. A sister yet orthogonal property of finite controllability is of paramount importance also in logic, where it has been investigated much earlier. It is known as *finite model property* or *finite satisfiability* (Ebbinghaus and Flum 1995), and it asks for a class $\mathscr{C}$ of sentences whether every satisfiable sentence of $\mathscr{C}$ has a finite model. For example, both Gödel and Schütte proved that $\forall^2\exists^*$ first-order sentences are finitely satisfiable.

Although reasoning under finite models has a long history and it has been actively investigated in various fields of Computer Science, finite controllability remains open for many languages combining or generalizing the key properties underlying the basic classes depicted in Figure 1. By way of example, we mention (i) glut-guarded (Krötzsch and Rudolph 2011) extending guarded and weakly-acyclic; (ii) weakly-sticky-join (Calì *et al.* 2012) extending sticky-join, weakly-acyclic, and shy; and (iii) tame (Gottlob *et al.* 2013) extending sticky and guarded.

Between guarded and glut-guarded, it is worth to recall weakly-guarded (Calì *et al.* 2013), where each rule body has an atom covering all those variables that only occur in invaded (a.k.a. affected) positions. Actually, this class is finitely controllable although the proof sketch given by Bárány *et al.* (2014) has some hole (there, some model of $D \cup \Sigma'$ might not satisfy Σ). In fact, our canonical rewriting yields an ontology that can be partitioned in *active* and *harmless*, where the active part is guarded. Well-supported models and propagation ordering behave as for shy.

An additional clarification concerns the notions of linear and sticky-join considered by Gogacz and Marcinkowski (2017), since they are not standard (actually stricter). In the former, repeated variables are admitted only in rule heads, while for the latter, the authors state that the difference between sticky and sticky-join "can only be seen if repeated variables in the heads of the rules are allowed". (Regarding sticky, the classical notion is only rephrased: their "immortal" positions correspond to positions being not marked.) From such a mismatch, however, it follows that finite controllability of sticky-join was unknown before our work. A curious reader may verify that the proof of their Lemma 4 breaks down when moving to a linear (hence, sticky-join) ontology such as $\Sigma = \{p(X,X) \rightarrow r(X);\ r(X) \rightarrow \exists Y\, r(Y)\}$—inducing no immortal position since all positions $p[1]$, $p[2]$ and $r[1]$ host marked variables—paired with the singleton database $D = \{p(c,c)\}$.

6 Conclusion

By demonstrating that shy is finitely controllable, we complete an important picture around the basic decidable Datalog$^{\pm}$ classes. But we take it as a starting point rather than an ending one.

On the one hand, finite controllability immediately implies decidability of OBQA. Actually, via the soundness and completeness of the chase procedure we know that the problem of deciding whether a UBCQ is *true* over a Datalog$^{\pm}$ theory is recursively enumerable. But the complementary problem of deciding whether a UBCQ is *false*

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

over a finitely controllable Datalog$^{\pm}$ class $\mathscr{C}$ is recursively enumerable too. In fact, each theory $D \cup \Sigma$, with $\Sigma \in \mathscr{C}$, always admits a fair lexicographic enumeration of its finite models. Unfortunately, such a naïve procedure would be inefficient in practice. Making it usable and competitive for real-world problems is challenging and it is part of our ongoing work. Basically, this would lead to a tool able to deal with any finitely controllable fragment, some of which (e.g., guarded) have no effective implementation.

On the other hand, we believe the techniques developed in this paper could have future applications. For example, we are working on an extended version of our canonical rewriting that encodes in the predicates also a limited amount of nulls. This requires more complex techniques, which however would apply to classes using the key properties underlying weakly-acyclic, such as glut-guarded and weakly-sticky-join (see Section 5). Hence, by combining these techniques with the above tool for finitely controllable classes, we aim at the design and implementation of a reasoner able to deal with ontologies falling in any known decidable Datalog$^{\pm}$ class.

Finally—even if the unrestricted set of existential rules cannot be finitely controllable since it is not decidable—it is still open, to the best of our knowledge, whether there exists, or not, a fragment of existential rules which is decidable but not finitely controllable.

Supplementary materials

To view supplementary material for this article, please visit https://doi.org/10.1017/S1471068417000369

References

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D. and Patel-Schneider, P. F., Eds. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications.* CUP.

Baget, J., Leclère, M. and Mugnier, M. 2010. Walking the decidability line for rules with existential variables. In *Proc. of KR'10.*

Baget, J., Leclère, M., Mugnier, M. and Salvat, E. 2009. Extending decidable cases for rules with existential variables. In *Proc. of IJCAI'09*, 677–682.

Baget, J., Leclère, M., Mugnier, M. and Salvat, E. 2011. On rules with existential variables: Walking the decidability line. *Artificial Intelligence Journal 175,* 9–10, 1620–1654.

Bárány, V., Gottlob, G. and Otto, M. 2014. Querying the guarded fragment. *Logical Methods in Computer Science 10,* 2, 1–35.

Bienvenu, M., ten Cate, B., Lutz, C. and Wolter, F. 2014. Ontology-based data access: A study through disjunctive datalog, csp, and MMSNP. *ACM TODS 39,* 4, 33:1–33:44.

Bourhis, P., Manna, M., Morak, M. and Pieris, A. 2016. Guarded-based disjunctive tuple-generating dependencies. *ACM TODS 41,* 4, 27:1–27:45.

Calì, A., Gottlob, G. and Kifer, M. 2013. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Artificial Intelligence Research (JAIR) 48*, 115–174.

Calì, A., Gottlob, G. and Lukasiewicz, T. 2009a. Datalog$^{\pm}$: A unified approach to ontologies and integrity constraints. In *Proc. of ICDT'09*, 14–30.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press

Calì, A., Gottlob, G. and Lukasiewicz, T. 2009b. Tractable query answering over ontologies with datalog+/−. In *Proc. of DL'09*.

Calì, A., Gottlob, G. and Lukasiewicz, T. 2012. A general datalog-based framework for tractable query answering over ontologies. *Journal of Web Semantics 14*, 57–83.

Calì, A., Gottlob, G. and Pieris, A. 2010. Advanced processing for ontological queries. *PVLDB 3,* 1, 554–565.

Calì, A., Gottlob, G. and Pieris, A. 2012. Towards more expressive ontology languages: The query answering problem. *Artificial Intelligence Journal 193*, 87–128.

Civili, C. and Rosati, R. 2012. A broad class of first-order rewritable tuple-generating dependencies. In *Proc. of Datalog 2.0*, 68–80.

Deutsch, A., Nash, A. and Remmel, J. B. 2008. The chase revisited. In *Proc. of PODS'08*, 149–158.

Ebbinghaus, H.-D. and Flum, J. 1995. *Satisfiability in the Finite*. Springer, Berlin Heidelberg, 95–103.

Fages, F. 1991. A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics. *New Generation Computing 9,* 3/4, 425–444.

Fagin, R., Kolaitis, P. G., Miller, R. J. and Popa, L. 2005. Data exchange: Semantics and query answering. *Theoretical Computer Science 336,* 1, 89–124.

Gogacz, T. and Marcinkowski, J. 2013. Converging to the chase – A tool for finite controllability. In *Proc. of LICS'13*, 540–549.

Gogacz, T. and Marcinkowski, J. 2017. Converging to the chase – A tool for finite controllability. *Journal of Computer and System Sciences 83,* 1, 180–206.

Gottlob, G., Kikot, S., Kontchakov, R., Podolskii, V. V., Schwentick, T. and Zakharyaschev, M. 2014. The price of query rewriting in ontology-based data access. *Artificial Intelligence Journal 213*, 42–59.

Gottlob, G., Manna, M. and Pieris, A. 2013. Combining decidability paradigms for existential rules. *Theory and Practice of Logic Programming 13,* 4–5, 877–892.

Gottlob, G., Orsi, G. and Pieris, A. 2014. Query rewriting and optimization for ontological databases. *ACM TODS 39,* 3, 25:1–25:46.

Gottlob, G., Pieris, A. and Tendera, L. 2013. Querying the guarded fragment with transitivity. In *Proc. of ICALP'13*, 287–298.

Johnson, D. S. and Klug, A. C. 1984. Testing containment of conjunctive queries under functional and inclusion dependencies. *Journal of Computer and System Sciences 28,* 1, 167–189.

Krötzsch, M. and Rudolph, S. 2011. Extending decidable existential rules by joining acyclicity and guardedness. In *Proc. of IJCAI'11*, 963–968.

Leone, N., Manna, M., Terracina, G. and Veltri, P. 2012. Efficiently computable datalog$^{\exists}$ programs. In *Proc. of KR'12*.

Rosati, R. 2006. On the decidability and finite controllability of query processing in databases with incomplete information. In *Proc. of PODS'06*.

Rosati, R. 2007. The limits of querying ontologies. In *Proc. of ICDT'07*, 164–178.

https://doi.org/10.1017/S1471068417000369 Published online by Cambridge University Press