Querying finite or arbitrary models? No matter! Existential rules may rely on both once again*

(discussion paper)

Giovanni Amendola[1], Nicola Leone[2], and Marco Manna[2]

[1] School of Informatics, University of Edinburgh, UK
gamendol@exseed.ed.ac.uk
[2] DeMaCS, University of Calabria, Italy
{leone,manna}@mat.unical.it

Abstract. Ontology-based query answering asks whether a Boolean query is satisfied by all models of a logical theory consisting of an extensional 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. 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 showing that finite controllability holds also for shy ontologies. The demonstration is based on a number of technical tools which could be used for similar purposes and are valuable per se.

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 [12, 17] to Database Theory [19, 5] and Logic [4, 20]. This problem, called *ontology-based query answering* (OBQA) [8], 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.

Description Logics [2] and Datalog$^{\pm}$ [7] have been recognized as the two main families of formal knowledge representation languages to specify Σ, while union of (Boolean) conjunctive queries, U(B)CQs for short, is the most common and studied formalism to express q. For both these families, OBQA is generally undecidable (see, [22] and [6], respectively). Hence, a number of syntactic decidable fragments of the above ontological languages have been singled out. However, decidability alone is not

* The paper has been partially supported by the 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)".

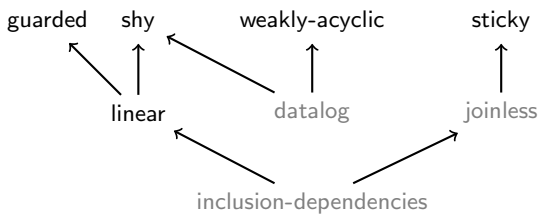

Fig. 1. Taxonomy of the basic Datalog$^{\pm}$ classes.

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* [23]. An ontological fragment $\mathcal{F}$ is said to be finitely controllable if, for each triple $\langle D, \Sigma, q \rangle$ with $\Sigma \in \mathcal{F}$, 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 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 [23, 4], 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" [9] 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 $\mathcal{EL}$ 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* [15], *guardedness* [6], *linearity* [9], *stickiness* [10], and *shyness* [21]. And these properties underlie the basic classes of existential rules called weakly-acyclic, guarded, linear, sticky, and shy, respectively. Several variants and combinations of these classes have been defined and studied too [11, 13, 18], as well as semantic properties subsuming the syntactic ones [3, 21].

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 collecting sets of 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 [16] 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.

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 ($\models_{\mathsf{fin}}$) is required, then the chase is generally uncomplete, unless ontologies are finitely controllable. Hence, proving this property is of utmost importance, especially in contexts where finite model reasoning is relevant, like for databases [23, 4].

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* [15]. 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 [23, 4], guarded [4], and more recently for sticky [16]. 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. *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) three key technical tools that we consider rather general and that we believe can be useful in the future for similar, or even more general, purposes. In particular, the first (canonical rewriting) and the third one (well-supported finite models) are applicable to every set of existential rules, besides shy ontologies. Finally, we exploit the fact that joinless sets of existential rules are finitely controllable [16]. (Full constructions and proofs are reported in [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* is a labeled tuple $p(t_1, \ldots, t_m)$, where p is a predicate symbol, m is the *arity*, and $t_1, \ldots, t_m$ are terms. An atom is *simple* if it contains no repeated term. 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, $h(A) = \{p(h(t_1), \ldots, h(t_m)) : p(t_1, \ldots, t_m) \in A\} \subseteq B$. 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. A *union of Boolean conjunctive query* (UBCQ) 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 called Boolean conjunctive query, or BCQ for short.

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. We recall the notion of *chase* [14]. Consider a logical theory $\langle D, \Sigma \rangle$ as above. A rule $\rho \in \Sigma$ is *applicable* to an instance I if there is a homomorphism h from $body(\rho)$ to I. Let $I' = I \cup h'(head(\rho))$, where $h' \supseteq h|_{\mathbf{X}}$ is a homomorphism from $head(\rho)$ to I' such that, for each $Z \in \mathbf{Z}$, $h'(Z)$ is a different fresh null not occurring in I. We will write $\langle \rho, h \rangle(I) = I'$. Indeed, it defines a single *chase step*. The *chase procedure* of $D \cup \Sigma$ is a sequence $I_0 = D \subset I_1 \subset I_2 \subset \ldots \subset I_m \subset \ldots$ of instances obtained by applying exhaustively the rules of Σ in a fair fashion such that, for each $i > 0$, $I_i = \langle \rho, h \rangle(I_{i-1})$, for some rule ρ and homomorphism h from $body(\rho)$ to I_{i-1}. We define $chase(D,\Sigma) = \cup_{i \geq 0} I_i$. It is well-known that $chase(D,\Sigma)$ is a *universal* model of $D \cup \Sigma$. In fact, 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 $chase(D,\Sigma) \models q$ if and only if $D \cup \Sigma \models q$ [15].

Shy ontologies. Consider a chase step $\langle \rho, h \rangle(I) = I'$ employed in the construction of $chase(D,\Sigma)$. If Σ is shy, then its underlying properties [21] guarantee that: (1) if X occurs in two different atoms of $body(\rho)$, then $h(X) \in \mathbf{C}$; and (2) if X and Y occur both in $head(\rho)$ and in two different atoms of $body(\rho)$, then $h(X) = h(Y)$ implies $h(X) \in \mathbf{C}$.

Finite controllability. The *finite models* of a theory $D \cup \Sigma$, denoted by $fmods(D,\Sigma)$, are defined as the set of instances $\{I \in mods(D,\Sigma) : |I| \in \mathbb{N}\}$, each of finite size. An ontological fragment $\mathcal{F}$ is said to be *finitely controllable* if, for each database D, for each ontology Σ of $\mathcal{F}$, 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 $D \cup \Sigma \models_{\mathsf{fin}} q$.

3 Technical Tools

3.1 Canonical Rewriting

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 containing only simple atoms; (4) $chase(D^c, \Sigma^c)$ is a constant-free instance containing only simple atoms; (5) there is a "semantic" bijection between $chase(D,\Sigma)$ and $chase(D^c, \Sigma^c)$. By exploiting the above five properties we are able to state the following result.

Theorem 2. *$D \cup \Sigma \models q$ if, and only if, $D^c \cup \Sigma^c \models q^c$.*

Let us now shed some light on our construction. To this aim, consider the ontology $\Sigma = \{person(X) \rightarrow \exists Y\, fatherOf(Y,X); fatherOf(X,Y) \rightarrow person(X)\}$, and the database $D = \{person(tim), person(john), fatherOf(tim, john)\}$. Let p, f, c_1 and c_2 be shorthands of *person*, *fatherOf*, *tim* and *john*, respectively. Hence, $chase(D,\Sigma)$ is $D \cup \{p(n_i)\}_{i>0} \cup \{f(n_1,c_1), f(n_2,c_2)\} \cup \{f(n_{i+2},n_i), f(n_{i+3},n_{i+1})\}_{i>0}$. From D we construct the propositional database $D^c = \{p_{[c_1]}, p_{[c_2]}, f_{[c_1,c_2]}\}$ obtained by encoding in the predicates the tuples of D. Then, from Σ we construct Σ^c collecting the following rules:

$$
\begin{array}{llll}
p_{[c_1]} \to \exists Y f_{[1,c_1]}(Y) & f_{[c_1,c_1]} \to p_{[c_1]} & f_{[c_1,1]}(Y) \to p_{[c_1]} & f_{[1,1]}(X) \to p_{[1]}(X) \\
p_{[c_2]} \to \exists Y f_{[1,c_2]}(Y) & f_{[c_1,c_2]} \to p_{[c_1]} & f_{[c_2,1]}(Y) \to p_{[c_2]} & f_{[1,2]}(X,Y) \to p_{[1]}(X) \\
p_{[1]}(X) \to \exists Y f_{[1,2]}(Y,X) & f_{[c_2,c_1]} \to p_{[c_2]} & f_{[1,c_1]}(X) \to p_{[1]}(X) & \\
 & f_{[c_2,c_2]} \to p_{[c_2]} & f_{[1,c_2]}(X) \to p_{[1]}(X) &
\end{array}
$$

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 $h(head(\rho^c)) = f_{[1,c_1]}(n_1)$. 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, $chase(D^c,\Sigma^c) = D^c \cup \{p_{[1]}(n_i)\}_{i>0} \cup \{f_{[1,c_1]}(n_1), f_{[1,c_2]}(n_2)\} \cup \{f_{[1,2]}(n_{i+2},n_i), f_{[1,2]}(n_{i+3},n_{i+1})\}_{i>0}$. 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 the BCQ $q = \exists X \exists Y r(X,Y) \wedge s(Y,c_1)$. Assume D contains only the constant c_1. Therefore, q^c is the UBCQ $\big(r_{[c_1,c_1]} \wedge s_{[c_1,c_1]}\big) \vee \big(\exists Y r_{[c_1,1]}(Y) \wedge s_{[1,c_1]}(Y)\big) \vee \big(\exists X r_{[1,c_1]}(X) \wedge s_{[c_1,c_1]}\big) \vee \big(\exists Y r_{[1,1]}(Y) \wedge s_{[1,c_1]}(Y)\big) \vee \big(\exists X \exists Y r_{[1,2]}(X,Y) \wedge s_{[1,c_1]}(Y)\big)$.

3.2 Active and Harmless Rules in Shy

Consider a pair $\langle D,\Sigma\rangle$ and the associated pair $\langle D^c,\Sigma^c\rangle$ in canonical form. The final goal of this section is to syntactically identify (and discard) rules of Σ^c that are never applied by the chase. But to be effective here, we need to exploit the key properties of Σ. So, in the rest of this section, let us assume that $\Sigma \in$ shy. First, we observe that the canonical rewriting of a shy ontology is indeed shy. Second, we partition Σ^c in two sets, denoted by Σ^c_a and Σ^c_h, collecting *active* and *harmless* rules, respectively, enjoying the following properties: (1) Σ^c_h are the rules of Σ^c with at least a variable occurring in more than one body atom; (2) $\Sigma^c_a = \Sigma^c \setminus \Sigma^c_h$ is a joinless (and still shy) ontology; and (3) $chase(D^c,\Sigma^c) = chase(D^c,\Sigma^c_a)$. By exploiting the above three properties we are able to state the following result.

Theorem 3. *For each $\Sigma \in$ shy, it holds that $D^c \cup \Sigma^c \models q^c$ if, and only if, $D^c \cup \Sigma^c_a \models q^c$.*

Also in this case, instead of proving formally the above theorem, we provide some insights regarding the way how we partition 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 collecting the following rules

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

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 second 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, which actually consider variable X as "protected" in $f(X,Y), p(X) \rightarrow p(Y)$. Hence, Σ^c_a contains only the joinless rules in the first column, while Σ^c_h contains the remaining "join-rules" in the second column.

3.3 Well-Supported Finite Models

Inspired by the notion of *well-supported interpretations* introduced in the context of general logic programming, we define the notion of *well-supported* finite models of a pair $\langle D, \Sigma \rangle$, denoted by $wsfmods(D,\Sigma)$, which enjoy the following properties: (1) for each $M \in fmods(D,\Sigma)$, there exists a well-supported finite model $M' \subseteq M$; (2) each minimal finite model of $D \cup \Sigma$ is a well-supported finite model. Assuming that the symbol $\models_{\mathsf{wsf}}$ refers to the satisfiability of the query under the well-supported finite models only, by exploiting the above two properties we are able to state the following result.

Theorem 4. *$D \cup \Sigma \models_{\mathsf{fin}} q$, if and only, if $D \cup \Sigma \models_{\mathsf{wsf}} q$.*

Unfortunately, in this case, we cannot avoid defining formally this notion. A finite instance I is called *well-supported* w.r.t. $D \cup \Sigma$ if there exists 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 two conditions is satisfied: (i) α_j is a database atom of D; or (ii) there exist a rule ρ of Σ and a homomorphism h from the atoms of ρ to $\{\alpha_1, \ldots, \alpha_j\}$ such that $h(head(\rho)) = \{\alpha_j\}$ and $h(body(\rho)) \subseteq \{\alpha_1, \ldots, \alpha_{j-1}\}$.

Interestingly, although each finite model of an ontological theory contains a well-supported finite model of the theory, the reverse inclusion does not hold, as shown in the following example. Consider the father-person ontology Σ given in Section 3.1. $M = D \cup \{f(c_1,c_1), f(c_2,c_1)\}$ is a well-supported finite model of $D \cup \Sigma$. Indeed, for instance, $(p(c_1), p(c_2), f(c_1,c_2), f(c_1,c_1), f(c_2,c_1))$ is a well-supported ordering of M. However, $M \setminus \{f(c_2,c_1)\}$ is a model of $D \cup \Sigma$. Therefore, M is not a minimal model.

4 Finite Controllability of Shy Ontologies

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

Theorem 5. *For each $\Sigma \in \mathsf{shy}$, it holds that $D \cup \Sigma \models_{\mathsf{fin}} q$ if, and only if, $D^c \cup \Sigma^c_a \models_{\mathsf{wsf}} q^c$.*

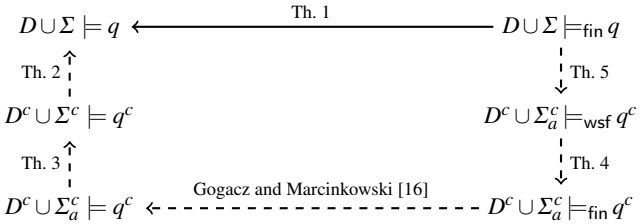

Fig. 2. Chain of implications for the proof of Theorem 1.

For our purposes, the "only if" direction is the most important one. Consider an arbitrary model $M^c \in \mathit{wsfmods}(D^c, \Sigma^c_a)$. Let $\mathit{decode}(M^c)$ be the instance obtained from M^c by decoding its atoms. For example, if $p_{[1,c,1]}(n_i) \in M^c$, then $p(n_i, c, n_i) \in \mathit{decode}(M^c)$. It suffices to prove that there exist $M \in \mathit{fmods}(D, \Sigma)$ and a homomorphism h' s.t. $h'(M) \subseteq \mathit{decode}(M^c)$. Indeed, by hypothesis, there exists a homomorphism h s.t. $h(q) \subseteq M$, and so $(h' \circ h)(q) \subseteq \mathit{decode}(M^c)$. Then, the result follows by the fact that $(h' \circ h)(q^c) \subseteq M^c$.

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

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

One can verify that $M^c = \{s_{[c]}, p_{[1]}(n_1), r_{[1]}(n_1)\}$ is a (minimal) well-supported finite model of $D^c \cup \Sigma^c_a$ since Σ^c_a is obtained from Σ^c by discarding the last harmless rule. However, $\mathit{decode}(M^c) = \{s(c), p(n_1), r(n_1)\}$ is not a model of $D \cup \Sigma$ because the third rule is not satisfied. The idea now is to show how to construct from $\mathit{decode}(M^c)$ a model $M \in \mathit{fmods}(D, \Sigma)$ that can be homomorphically mapped to $\mathit{decode}(M^c)$. Intuitively, we identify the *starting points* in which existentially quantified variables of Σ^c_a have been satisfied and rename the introduced terms using the so-called *propagation ordering*. In the example above, consider the ordering $(s(c), p(n_1), r(n_1))$ of $\mathit{decode}(M^c)$, replace n_1 in $p(n_1)$ by $\langle n_1, 2, 2\rangle$ (null n_1 introduced in the second atom in the second position), and replace n_1 in $r(n_1)$ by $\langle n_1, 3, 2\rangle$ (null n_1 introduced in the third atom in the second position). Then, since M^c is well-supported, we propagate (if needed) these new terms according the supporting ordering. In our case, $M = \{s(c), p(\langle n_1, 2, 2\rangle), r(\langle n_1, 3, 2\rangle)\}$ is now a finite model of $D \cup \Sigma$ that can be mapped to $\mathit{decode}(M^c)$.

Finally, to prove Theorem 1, we can now combine the trivial "only if" implication with the dashed chain of intermediate results given in Figure 2.

5 Conclusion

At this point, also thanks to the results shown in this paper, we know that every basic decidable Datalog$^{\pm}$ class is finitely controllable. However, the problem remains open for extensions and combinations of these classes [11, 13, 18]. For example, it is open whether finite controllability holds for the following classes: (*i*) sticky-join, generalizing sticky and linear; (*ii*) tame, generalizing sticky and guarded; and (*iii*) weakly-

sticky-join, generalizing sticky-join, weakly-acyclic and shy. We believe that the techniques developed in this paper can be exploited to give an answer to these questions.

References

1. Amendola, G., Leone, N., Manna, M.: Finite model reasoning over existential rules. Forthcoming (2017)
2. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. CUP (2003)
3. Baget, J., Leclère, M., Mugnier, M., Salvat, E.: Extending decidable cases for rules with existential variables. In: In Proc. of IJCAI'09. pp. 677–682 (2009)
4. Bárány, V., Gottlob, G., Otto, M.: Querying the guarded fragment. Logical Methods in Computer Science 10(2) (2014)
5. Bourhis, P., Manna, M., Morak, M., Pieris, A.: Guarded-based disjunctive tuple-generating dependencies. ACM TODS 41(4), 27:1–27:45 (2016)
6. Calì, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. J. Artif. Intell. Res. (JAIR) 48, 115–174 (2013)
7. Calì, A., Gottlob, G., Lukasiewicz, T.: Datalog$^{\pm}$: a unified approach to ontologies and integrity constraints. In: In Proc. of ICDT'09. pp. 14–30 (2009)
8. Calì, A., Gottlob, G., Lukasiewicz, T.: Tractable query answering over ontologies with datalog+/-. In: In Proc. of DL'09 (2009)
9. Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. J. Web Sem. 14, 57–83 (2012)
10. Calì, A., Gottlob, G., Pieris, A.: Advanced processing for ontological queries. PVLDB 3(1), 554–565 (2010)
11. Calì, A., Gottlob, G., Pieris, A.: Towards more expressive ontology languages: The query answering problem. AIJ 193, 87–128 (2012)
12. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. AIJ 195, 335–360 (2013)
13. Civili, C., Rosati, R.: A broad class of first-order rewritable tuple-generating dependencies. In: In Proc. of Datalog 2.0. pp. 68–80 (2012)
14. Deutsch, A., Nash, A., Remmel, J.B.: The chase revisited. In: In Proc. of PODS'08. pp. 149–158 (2008)
15. Fagin, R., Kolaitis, P.G., Miller, R.J., Popa, L.: Data exchange: semantics and query answering. TCS 336(1), 89–124 (2005)
16. Gogacz, T., Marcinkowski, J.: Converging to the chase - A tool for finite controllability. In: In Proc. of LICS'13. pp. 540–549 (2013)
17. Gottlob, G., Kikot, S., Kontchakov, R., Podolskii, V.V., Schwentick, T., Zakharyaschev, M.: The price of query rewriting in ontology-based data access. AIJ 213, 42–59 (2014)
18. Gottlob, G., Manna, M., Pieris, A.: Combining decidability paradigms for existential rules. TPLP 13(4-5), 877–892 (2013)
19. Gottlob, G., Orsi, G., Pieris, A.: Query rewriting and optimization for ontological databases. ACM TODS 39(3), 25:1–25:46 (2014)
20. Gottlob, G., Pieris, A., Tendera, L.: Querying the guarded fragment with transitivity. In: In Proc. of ICALP'13. pp. 287–298 (2013)
21. Leone, N., Manna, M., Terracina, G., Veltri, P.: Efficiently computable Datalog$^{\exists}$ programs. In: In Proc. of KR'12 (2012)
22. Rosati, R.: The limits of querying ontologies. In: In Proc. of ICDT'07. pp. 164–178 (2007)
23. Rosati, R.: On the finite controllability of conjunctive query answering in databases under open-world assumption. JCSS 77(3), 572–594 (2011)