Online appendix for the paper

Combining decidability paradigms for existential rules

published in Theory and Practice of Logic Programming

GEORG GOTTLOB
Department of Computer Science, University of Oxford, UK
(*e-mail:* `georg.gottlob@cs.ox.ac.uk`)

MARCO MANNA
Department of Mathematics and Informatics, University of Calabria, Italy
(*e-mail:* `manna@mat.unical.it`)

ANDREAS PIERIS
Department of Computer Science, University of Oxford, UK
(*e-mail:* `andreas.pieris@cs.ox.ac.uk`)

submitted 10 April 2013; revised 23 May 2013; accepted 23 June 2013

Appendix A Definitions and Background

The TGD Chase Procedure

A *chase sequence* of a database D w.r.t. a set Σ of TGDs is a sequence of chase steps $I_i\langle\sigma_i,h_i\rangle I_{i+1}$, where $i \geqslant 0$, $I_0 = D$ and $\sigma_i \in \Sigma$. The chase of D w.r.t. Σ, denoted $chase(D,\Sigma)$, is defined as follows:

– A *finite chase* of D w.r.t. Σ is a finite chase sequence $I_i\langle\sigma_i,h_i\rangle I_{i+1}$, where $0 \leqslant i < m$, and there is no $\sigma \in \Sigma$ which is applicable to I_m; let $chase(D,\Sigma) = I_m$.
– An infinite chase sequence $I_i\langle\sigma_i,h_i\rangle I_{i+1}$, where $i \geqslant 0$, is *fair* if whenever a TGD σ : $\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists\mathbf{Z}\,\psi(\mathbf{X},\mathbf{Z})$ of Σ is applicable to I_i with homomorphism h, then there exists $h' \supseteq h$ and $k > i$ such that $h'(head(\sigma)) \subseteq I_k$. An *infinite chase* of D w.r.t. Σ is a fair infinite chase sequence $I_i\langle\sigma_i,h_i\rangle I_{i+1}$, where $i \geqslant 0$; let $chase(D,\Sigma) = \bigcup_{i=0}^{\infty} I_i$.

Example Appendix A.1
Consider the set Σ constituted by the TGDs

$$\sigma_1 \ : \ r(X,Y,Z) \rightarrow s(Y,X) \qquad\qquad \sigma_2 \ : \ s(X,Y) \rightarrow \exists Z \exists W\, r(Y,Z,W),$$

and let $D = \{r(a,b,c)\}$. An infinite chase of D w.r.t. Σ is:

$$\begin{gathered} D \\ \langle\sigma_1, h_1 = \{X \rightarrow a, Y \rightarrow b, Z \rightarrow c\}\rangle \\ D \cup \{s(b,a)\} \\ \langle\sigma_2, h_2 = \{X \rightarrow b, Y \rightarrow a\}\rangle \\ D \cup \{s(b,a), r(a,z_1,z_2)\} \\ \langle\sigma_1, h_3 = \{X \rightarrow a, Y \rightarrow z_1, Z \rightarrow z_2\}\rangle \end{gathered}$$

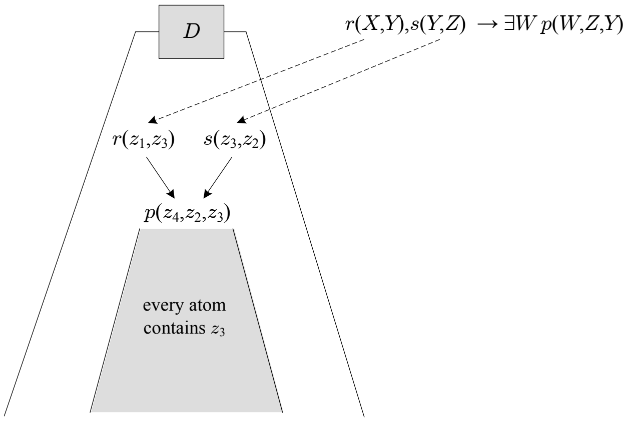

Fig. A 1. Sticky property.

$$D \cup \{s(b,a), r(a,z_1,z_2), s(z_1,a)\}$$
$$\vdots$$
$$\langle \sigma_2, h_{2i+2} = \{X \to z_{2i-1}, Y \to a\}\rangle$$
$$D \cup \{s(b,a), r(a,z_1,z_2)\} \cup \bigcup_{j=1}^{i} \{s(z_{2j-1},a), r(a,z_{2j+1},z_{2j+2})\}$$
$$\vdots$$

Clearly, $chase(D,\Sigma)$ is the infinite instance

$$\{r(a,b,c), s(b,a), r(a,z_1,z_2)\} \cup \bigcup_{j=1}^{\infty} \{s(z_{2j-1},a), r(a,z_{2j+1},z_{2j+2})\},$$

where $z_1, z_2, \ldots$ are nulls of Γ_N. ∎

The Sticky Property

It is interesting to see that the chase constructed under a sticky set of TGDs enjoys a syntactic property called the *sticky property*.

Definition Appendix A.1 (*Sticky Property*)
Consider a database D for a schema $\mathscr{R}$, and a set Σ of TGDs over $\mathscr{R}$. Suppose that the chase step $chase^{[k-1]}(D,\Sigma)\langle\sigma,h\rangle chase^{[k]}(D,\Sigma)$, where $k \geqslant 1$, is applied during the construction of $chase(D,\Sigma)$. Then, $chase(D,\Sigma)$ is *k-sticky* if, for each variable V that occurs in $body(\sigma)$ more than once, and for each $\underline{a} \in (chase^{[k]}(D,\Sigma) \setminus chase^{[k-1]}(D,\Sigma))$, $h(V)$ occurs in $\underline{a}$, and also in every atom $\underline{b}$ such that $\langle \underline{a}, \underline{b}\rangle$ belongs to $CR^{+}[D,\Sigma]$. We say that the chase of D w.r.t. Σ has the *sticky property* if $chase(D,\Sigma)$ is k-sticky, for each $k \geqslant 1$. ∎

Generally speaking, the sticky property imposes the following condition: the symbols which are associated (during the application of a TGD σ) with the body-variables of σ that occur more than once, appear in the generated atom $\underline{a}$, and also in every atom obtained from some chase derivation which involves $\underline{a}$, thus “sticking” to all such atoms.

Example Appendix A.2
Consider a database D and a set Σ which contains (among others) the TGD $\sigma : r(X,Y), s(Y,Z) \to \exists W\, p(W,Z,Y)$. Suppose that the atoms $r(z_1,z_3)$ and $s(z_3,z_2)$, where z_1, z_2 and z_3 are nulls, occur

in the chase of D w.r.t. Σ. Clearly, σ is triggered and the atom $p(z_4,z_2,z_3)$, where z_4 is a null, is generated. The sticky property requires the null z_3, which is associated to the variable Y that occurs more than once in the body of σ, to appear in $p(z_4,z_2,z_3)$, and also in every atom obtained from some chase derivation which involves $p(z_4,z_2,z_3)$; see Figure A 1. ∎

As shown in (Calì et al. 2012), stickiness is a sufficient condition for the sticky property of the chase. In other words, given a set $\Sigma \in \mathsf{sticky}$ over a schema $\mathscr{R}$, $chase(D,\Sigma)$ enjoys the sticky property, for every database D for $\mathscr{R}$.

Appendix B Tameness

Theorem 3.1
BCQ answering is undecidable under: (1) $\mathsf{linear}|\mathsf{sticky}$, even for a single linear TGD and a single sticky TGD, and (2)$\mathsf{linear}|\mathsf{sticky}$, even for sticky rules where each variable occurs only once.

Proof
Part 1. It is well-known that there exists a single TGD σ such that BCQ answering under $\{\sigma\}$ is already undecidable (Baget et al. 2011). It is easy to transform σ into an equivalent (for query answering purposes) set Σ constituted by a single linear rule and a single sticky rule. In particular, assuming that $\{X_1,\ldots,X_k\}$ is the set of body-variables of σ, Σ is constituted by the sticky rule $body(\sigma) \rightarrow p(X_1,\ldots,X_k)$, where p is an auxiliary n-ary predicate, and the linear rule $p(X_1,\ldots,X_k) \rightarrow head(\sigma)$.

Part 2. The proof is by reduction from the problem of BCQ answering under arbitrary TGDs. Consider a set Σ of TGDs. For each $\sigma \in \Sigma$, if $\{X_1,\ldots,X_k\}$ is the set of the body-variables of σ, and n_i is the number of occurrences of X_i in $body(\sigma)$, then let

$$\tau_1(\sigma) = p_\sigma(\underbrace{X_1,\ldots,X_1}_{n_1},\ldots,\underbrace{X_k,\ldots,X_k}_{n_k}) \rightarrow head(\sigma),$$
$$\tau_2(\sigma) = \overline{body(\sigma)} \rightarrow p_\sigma(X_1^1,\ldots,X_1^{n_1},\ldots,X_k^1,\ldots,X_k^{n_k}),$$

where $\overline{body(\sigma)}$ is obtained from $body(\sigma)$ by replacing the j-th occurrence of X_i with X_i^j, and p_σ is an auxiliary predicate. Let $\Sigma_i = \{\tau_i(\sigma) \mid \sigma \in \Sigma\}_{i\in\{1,2\}}$. Finally, let $\Sigma' = \Sigma_1 \cup \Sigma_2$. It is easy to verify that $\Sigma_1 \in \mathsf{linear}$ and $\Sigma_2 \in \mathsf{sticky}$; thus, $\Sigma' \in \mathsf{linear}|\mathsf{sticky}$. Observe that, except for the atoms with an auxiliary predicate, $chase(D,\Sigma)$ and $chase(D,\Sigma')$ coincide, for each database D. Since the auxiliary predicates do not match any predicate symbol in any BCQ q, $chase(D,\Sigma) \models q$ iff $chase(D,\Sigma') \models q$, and the claim follows. □

Proposition 3.1
The problem of deciding whether a set $\Sigma \in \mathsf{guarded}|\mathsf{sticky}$ is predicate-tame is in PTIME.

Proof
Let $\mathscr{R}$ be the set of predicates occurring in Σ, and $cover(\sigma)$, where $\sigma \in \Sigma$, be the set of atoms of $body(\sigma)$ which contain all the body-variables. The *unprotected* predicates of $\mathscr{R}$ are defined inductively as follows. A predicate r is unprotected if there exists $\sigma \in \Sigma$ such that $cover(\sigma) = \varnothing$ and $r \in pred(head(\sigma))$. Moreover, for each $\sigma \in \Sigma$ such that all the predicates occurring in $pred(cover(\sigma))$ are unprotected, each predicate of $pred(head(\sigma))$ is unprotected. A predicate of $\mathscr{R}$ is *protected* if it is not unprotected; let $\mathscr{R}_p$ be the set of protected predicates of $\mathscr{R}$. The protected part of Σ, denoted Σ_p, is defined as the set $\{\sigma \mid \sigma \in \Sigma$ and there exists $\underline{a} \in cover(\sigma)$ such that $pred(\underline{a}) \in \mathscr{R}_p\}$, while the unprotected part of Σ, denoted Σ_u, is the set $\Sigma \setminus \Sigma_p$.

The next auxiliary lemma shows the connection between predicate-tameness and the unprotected part of a set of TGDs.

Lemma Appendix B.1
Σ is predicate-tame iff $\Sigma_u = \emptyset$ or $\Sigma_u \in \mathsf{sticky}$.

Proof
($\Rightarrow$) Assume first that Σ is predicate-tamed. By definition, there exists $\{\Sigma_g, \Sigma_s\} \in P_{\mathsf{guarded}|\mathfrak{C}}(\Sigma)$ and a guard function $g \in \mathit{Guard}(\Sigma_g)$ such that, for each $\sigma \in \Sigma_s$, there is no $\sigma' \in \Sigma_g$ for which $\mathit{pred}(g(\sigma')) \in \mathit{pred}(\mathit{head}(\sigma))$. Observe that each subset of a set of $\mathfrak{C}$ is still in $\mathfrak{C}$; this holds since, by removing a TGD from a sticky (resp., shy) set of TGDs, stickiness (resp., shyness) is preserved. Hence, it suffices to show that $\Sigma_u \subseteq \Sigma_s$. By contradiction, assume that $\Sigma_u \not\subseteq \Sigma_s$. This implies that there exists $\sigma \in \Sigma$ such that $\sigma \in \Sigma_u$ and $\sigma \notin \Sigma_s$. Since $\Sigma_g = \Sigma \setminus \Sigma_s$, $\sigma \in \Sigma_g$. Therefore, σ is a guarded TGD such that, for each $\sigma' \in \Sigma_s$, $\mathit{pred}(g(\sigma)) \notin \mathit{pred}(\mathit{head}(\sigma'))$. It is not difficult to see that $g(\sigma) \in \mathit{cover}(\sigma)$ and $\mathit{pred}(g(\sigma)) \in \mathscr{R}_p$, and thus $\sigma \in \Sigma_p$. But, since $\Sigma_p \cap \Sigma_u = \emptyset$, this contradicts the fact that $\sigma \in \Sigma_u$.

($\Leftarrow$) If $\Sigma_u = \emptyset$, then $\Sigma \in \mathsf{guarded}$, and the claim follows immediately. Suppose now that $\Sigma_u \in \mathfrak{C}$. It is easy to see that $\{\Sigma_p, \Sigma_u\} \in P_{\mathsf{guarded}|\mathfrak{C}}(\Sigma)$. Moreover, a guard function $g \in \mathit{Guard}(\Sigma_p)$, which satisfies the condition of Definition 3.4, can be constructed. In particular, for each $\sigma \in \Sigma_p$, let $g(\sigma)$ be an arbitrary atom of the (non-empty) set $\mathit{cover}(\sigma)$. Consequently, Σ is predicate-tamed, and the claim follows. $\square$

Clearly, the unprotected part of Σ can be constructed in polynomial time. The claim follows from the above lemma and the fact that the problem of deciding whether a set of TGDs belongs to sticky is in PTIME (Calì et al. 2012). The latter follows by observing that at each application of the propagation step at least one body-variable is marked; thus, after polynomially many steps the SMarking procedure terminates. $\square$

Appendix C Querying the Tame Fragment

Technical Results

Lemma 4.1
It holds that, $S_{in} \cap S_{out} \subseteq \mathit{terms}(\underline{a})$.

Proof
Towards a contradiction, assume that $S_{in} \cap S_{out}$ contains a term $t \notin \mathit{terms}(\underline{a})$. Due to guardedness and by Definition 4.3, t is a null. This fact immediately implies that there exists an atom $\underline{b} \in \mathit{reach}_{gs}(\underline{a}, I, \Sigma)$, generated from a rule of Σ_g, in which t is invented, and also implies that there exists an atom $\underline{c} \in \mathit{atype}(\underline{a}, I, \Sigma)$ such that $t \in \mathit{terms}(\underline{c})$. Hence, $\underline{c}$ belongs to $\mathit{reach}(\underline{b}, I, \Sigma)$. Since $\underline{b}$ is reachable from $\underline{a}$, it also holds that $\underline{c} \in \mathit{reach}(\underline{a}, I, \Sigma)$. This means, by Definition 4.2, that $\underline{c} \in \mathit{reach}(\underline{a}, I, \Sigma) \setminus \mathit{reach}_t(\underline{a}, I, \Sigma)$. Moreover, by Definition 4.1, $\underline{c}$ is invented from a rule of Σ_g and has a parent, say $\underline{d}$, which is not reachable from $\underline{a}$. However, this immediately implies that $\underline{d}$ is not reachable from $\underline{b}$ and, therefore, that $t \notin \mathit{terms}(\underline{d})$. But this is not possible because $\underline{c}$ is generated from a guarded rule and t is not invented in $\underline{c}$. Thus, t does not occur in S_{out} which is a contradiction, and the claim follows. $\square$

Lemma Appendix C.1
Consider a BCQ q over a schema $\mathscr{R}$, a database D for $\mathscr{R}$, and a set Σ of TGDs over $\mathscr{R}$. A set Σ' of TGDs over a schema $\mathscr{R}'$, where each TGD has only one head-atom with at most one existentially quantified variable which occurs once, can be constructed in LOGSPACE such that $D \cup \Sigma \models q$ iff $D \cup \Sigma' \models q$. Moreover, if Σ is tame, then Σ' is also tame.

Proof
We obtain Σ' from Σ by applying the following: for each $\sigma \in \Sigma$, if σ is already in the desired syntactic form, then $\tau(\sigma) = \{\sigma\}$; otherwise, assuming that $\{\underline{a}_1, \ldots, \underline{a}_k\} = head(\sigma)$, $\{X_1, \ldots, X_n\} = var(body(\sigma)) \cap var(head(\sigma))$, and $Z_1, \ldots, Z_m$ are the existentially quantified variables of σ, let $\tau(\sigma)$ be the set

$$\begin{array}{rcl} body(\sigma) & \rightarrow & \exists Z_1\, p^1_\sigma(X_1,\ldots,X_n,Z_1) \\ p^1_\sigma(X_1,\ldots,X_n,Z_1) & \rightarrow & \exists Z_2\, p^2_\sigma(X_1,\ldots,X_n,Z_1,Z_2) \\ p^2_\sigma(X_1,\ldots,X_n,Z_1,Z_2) & \rightarrow & \exists Z_3\, p^3_\sigma(X_1,\ldots,X_n,Z_1,Z_2,Z_3) \\ & \vdots & \\ p^{m-1}_\sigma(X_1,\ldots,X_n,Z_1,\ldots,Z_{m-1}) & \rightarrow & \exists Z_m\, p^m_\sigma(X_1,\ldots,X_n,Z_1,\ldots,Z_m) \\ p^m_\sigma(X_1,\ldots,X_n,Z_1,\ldots,Z_m) & \rightarrow & \underline{a}_1 \\ & \vdots & \\ p^m_\sigma(X_1,\ldots,X_n,Z_1,\ldots,Z_m) & \rightarrow & \underline{a}_k, \end{array}$$

where p^i_σ is an $(n+i)$-ary auxiliary predicate not occurring in $\mathscr{R}$, for each $i \in \{1, \ldots, m\}$. Let $\Sigma' = \bigcup_{\sigma \in \Sigma} \tau(\sigma)$, and $\mathscr{R}'$ be the schema obtained by adding to $\mathscr{R}$ the auxiliary predicates introduced above. It is easy to see that Σ' can be constructed in LOGSPACE. The auxiliary predicates, being introduced only during the above construction, do not match any predicate symbol in q, and hence $chase(D,\Sigma) \models q$ iff $chase(D,\Sigma') \models q$, or, equivalently, $D \cup \Sigma \models q$ iff $D \cup \Sigma' \models q$. The fact that, if Σ is tame, then Σ' is also tame, can be established easily and we leave the proof as a simple exercise to the interested reader. $\square$

The Algorithm TameQAns

The first step of the algorithm guesses some auxiliary structures that will be used in the rest of the execution. More precisely, the algorithm first guesses an image of the query q, namely a homomorphism h that maps q into $chase(D,\Sigma)$; for convenience, $h(q)$ is stored in Q, while the set of nulls occurring in $terms(Q)$ is stored in N. Next, TameQAns guesses a partition $\{N_g, N_s\}$ of N, where N_g are the nulls invented by guarded rules, while N_s are the ones invented by sticky rules. Then, the algorithm guesses a poset P on N_g, with M being the minimal elements of P, and each element of M is the root of a rooted tree. Finally, the shapes of the atoms where the nulls of Q are invented are guessed, and also, for every atom where a null of N_g is invented, a finite segment of its active type is guessed. We now proceed with the following universal steps:

Guarded Resolution Steps. The atoms in which the minimal elements of P are invented, as well as their (finite) active type, can be proved independently. This is done, for each minimal element of P, in a universal branch of the computation of TameQAns. Notice that, for a minimal element z, it is not enough to consider $\underline{a}_z$ together with the guessed finite segment $T(\underline{a}_z)$ of its active type as a new query and prove it independently by starting again TameQAns, since the information that z has been invented in $\underline{a}_z$ may be lost. Therefore, the algorithm performs

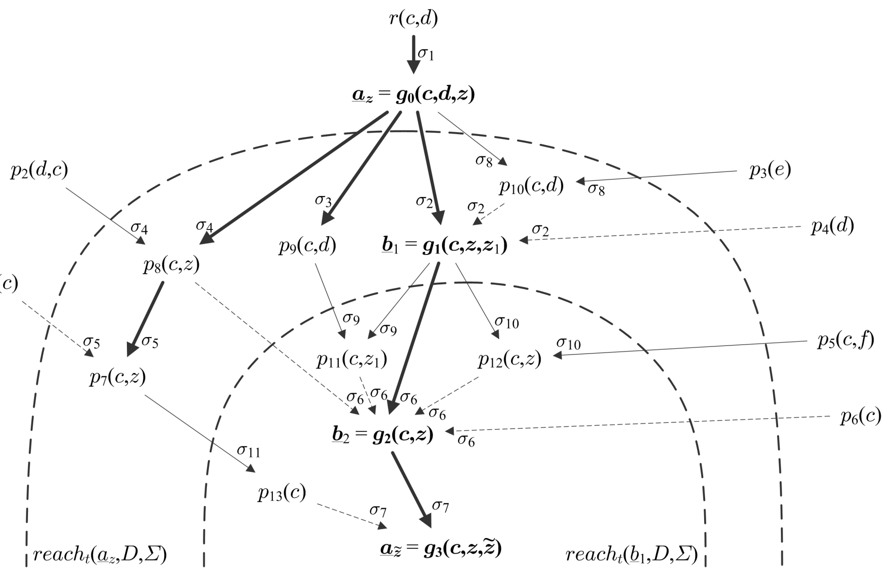

Fig. C 1. Guarded chase steps.

a resolution step via a guarded rule in order to bypass the chase step where $\underline{a}_z$ is generated. We assume that a most general unifier is always the identity on N. After this resolution step, the obtained set of atoms $\theta(body(\sigma))$ together with $T(\underline{a}_z)$ are considered as a new conjunctive query which can be now proved independently.

Guarded Chase Steps. For each pair $z \dot{\longrightarrow} \tilde{z} \in P$, the algorithm attempts to reach $\underline{a}_{\tilde{z}}$ starting from $\underline{a}_z$ by applying guarded chase steps. During each chase step, it generates the child of $\underline{a}_{\tilde{z}}$, say $\underline{b}$, by trusting the finite active type $T(\underline{a}_z)$, guesses a finite active type $T(\underline{b})$ for $\underline{b}$, and assigns to Q the side atoms that have been used to generate $\underline{b}$ together with $T(\underline{b})$, but without the atoms that are already in $T(\underline{a}_z)$ since we do not need to prove them again. Then, $\underline{a}_z$ and $T(\underline{a}_z)$ are replaced, respectively, by $\underline{b}$ and $T(\underline{b})$. Finally, in universal branches of the computation (at step 13), the algorithm proceeds with a chase step in order to reach $\underline{a}_{\tilde{z}}$ (step 7), and also proves Q (step 14). An example which explains the above description follows.

Example Appendix C.1

Let Σ be the tame set of TGDs:

$$
\begin{array}{rcl}
\sigma_1 & : & r(X,Y) \to \exists Z\, g_0(X,Y,Z) \\
\sigma_2 & : & g_0(X,Y,Z), p_{10}(X,Y), p_4(Y) \to \exists W\, g_1(X,Z,W) \\
\sigma_3 & : & g_0(X,Y,Z) \to p_9(X,Y) \\
\sigma_4 & : & g_0(X,Y,Z), p_2(Y,X) \to p_8(X,Z) \\
\sigma_5 & : & p_8(X,Y), p_1(X) \to p_7(X,Y) \\
\sigma_6 & : & g_1(X,Y,Z), p_8(X,Y), p_{11}(X,Z), p_{12}(X,Y), p_6(X) \to g_2(X,Y) \\
\sigma_7 & : & g_2(X,Y), p_{13}(X) \to \exists Z\, g_3(X,Y,Z) \\
\sigma_8 & : & g_0(X,Y,Z), p_3(W), \to p_{10}(X,Y) \\
\sigma_9 & : & g_1(X,Y,Z), p_9(X,W) \to p_{11}(X,Z) \\
\sigma_{10} & : & g_1(X,Y,Z), p_5(X,W) \to p_{12}(X,Y) \\
\sigma_{11} & : & p_7(X,Y) \to p_{13}(X),
\end{array}
$$

where $\Sigma_s = \{\sigma_8, \ldots, \sigma_{11}\}$ and $\Sigma_g = \Sigma \setminus \Sigma_s$, and $D = \{r(c,d),\ p_1(c),\ p_2(d,c),\ p_3(e),\ p_4(d),\ p_5(c,f),\ p_6(c)\}$. The chase of D and Σ is depicted in Figure C 1. As in Figure 2, bold and continuous arrows denote guarded and sticky chase derivations, respectively; dashed arrows denote the contribution from side atoms in guarded derivations only. Our intention is to show how to reach $\underline{a}_{\bar{z}}$ from $\underline{a}_z$ and its (finite) active type. The active type $T(\underline{a}_z)$ of $\underline{a}_z$ is $D \setminus \{r(c,d)\} \cup \{\underline{a}_z\}$, while the active type $T(\underline{b}_1)$ of $\underline{b}_1$ is the set $\{p_5(c,f),\ p_6(c),\ p_7(c,z),\ p_8(c,z),\ p_9(c,d)\} \cup \{\underline{b}_1\}$. The application of σ_2 requires the side atoms $p_4(d)$ and $p_{10}(c,d)$. The former is in the active type of $\underline{a}_z$; the latter, even if it is not in $T(\underline{a}_z)$, it belongs to $reach_{gs}(\underline{a}_z, D, \Sigma)$ and, therefore, it can be proven from the pair $\underline{a}_z$ and $T(\underline{a}_z)$. The application of σ_6 requires the side atoms $p_6(c)$, $p_8(c,z)$, $p_{11}(c,z_1)$ and $p_{12}(c,z)$. The first two atoms are in the active type of $T(\underline{b}_1)$, while the remaining two atoms belong to $reach_{gs}(\underline{b}_1, D, \Sigma)$. Since we trust $T(\underline{a}_z)$, to apply σ_2 and to build $T(\underline{b}_1)$ we need to prove only the atoms $p_7(c,z)$, $p_8(c,z)$ $p_9(c,d)$ and $p_{10}(c,d)$. These atoms, by Proposition 4.1, can be proven from the pair $\langle \underline{a}_z, T(\underline{a}_z)\rangle$ (step 13). ■

Hybrid Steps. For each atom $\underline{a} \in Q$ in which no null of N_g is invented, TameQAns applies resolution steps (via sticky rules) to reach either the given database, or some active type, or some atom generated from a guarded rule. In the first and second case, the algorithm proves $\underline{a}$ by resolution steps via sticky rules, without involving any guarded rule. In the third case, the algorithm transforms $\underline{a}$ in a number of atoms that are generated by rules of Σ_g, which in turn have to be proved forward from an atom also generated by a guarded rule and its finite active type. (This is the case, for example, of $s_5(z_5,z_4)$ and $s_4(y_4,z_5,z_7,z_6)$ that are first transformed in some anonymous atoms, which in turn are proved from $g_4(z_4,z_3,\ldots)$ and $g_5(z_5,z_3,z_1,\ldots)$).

Theorem 4.1

BCQ answering under tame sets of TGDs is 2EXPTIME-complete in combined complexity, EXPTIME-complete in case of bounded arity, NP-complete in case of fixed TGDs, and PTIME-complete in data complexity.

Proof

Consider a (non-normalized) tame set Σ over a schema $\mathscr{R}$, and let Σ' be the normalized version of Σ. Let w be the maximum arity over all predicates occurring in $\mathscr{R}$, and w' be the maximum arity over all auxiliary predicates, introduced during the normalization procedure, occurring in Σ'; in general, $w \leqslant w'$. During the construction of the chase under Σ', none of the side atoms of a guarded rule can be mapped to an atom with an auxiliary predicate. Moreover, none of the atoms of a sticky rule, with at least two body-atoms, can be mapped to an atom with an auxiliary predicate. Therefore, none of the auxiliary predicates have outgoing arrows crossing the boundary of $reach_t(\underline{a}, D, \Sigma)$. Consequently, the size of a (finite) active type T (as identified in the proof of Proposition 4.1) is at most $|\mathscr{R}| \cdot (w+1)^w$. In fact, by construction, we have that $|T| \leq |T_\star|$, and that $|T_\star| \leq |\mathscr{R}| \cdot (w+1)^w$, where $w+1$ comes from the fact that we need to consider the terms of $terms(\underline{a})$ plus the symbol $\star$. At each step of the computation, the alternating algorithm TameQAns needs to remember, for each variable in the query, at most one type. The upper bounds follow since AEXPSPACE = 2EXPTIME, APSPACE = EXPTIME and ALOGSPACE = PTIME, and from the fact that the number of different terms occurring in T are no more than $w \cdot |\mathscr{R}| \cdot (w+1)^w$. Actually, assuming that q contains n variables, TameQAns can be equipped with a finite subset of Γ_N having size $n \cdot w \cdot |\mathscr{R}| \cdot (w+1)^w + 1$, since at each step of the computation we can reuse symbols that in other steps (or branches) of the computation have a different semantic meaning. Notice that in the case of a fixed set of TGDs we need to perform a nondeterministic guessing in polynomial time (step 1), and then prove the minimal elements of M, the pairs of P, and the atoms of $Q \setminus \{\underline{a}_z \mid z \in N_g\}$, in a sequential manner by calling a PTIME oracle (which is our algorithm). Therefore, the overall procedure runs in $\text{NP}^{\text{PTIME}} = \text{NP}$. The lower bounds are inherited from BCQ answering under guarded TGDs (Calì et al. 2008). $\square$