Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Conservative Extensions for Existential Rules

Jean Christoph Jung[1] , **Carsten Lutz**[2] , **Jerzy Marcinkowski**[3]
[1]Institute of Computer Science, University of Hildesheim, Germany
[2]Institute of Computer Science, Leipzig University, Germany
[3]Institute of Computer Science, University of Wrocław, Poland
jungj@uni-hildesheim.de, clu@informatik.uni-leipzig.de, jma@cs.uni.wroc.pl

Abstract

We study the problem to decide, given sets T_1, T_2 of tuple-generating dependencies (TGDs), also called existential rules, whether T_2 is a conservative extension of T_1. We consider two natural notions of conservative extension, one pertaining to answers to conjunctive queries over databases and one to homomorphisms between chased databases. Our main results are that these problems are undecidable for linear TGDs, undecidable for guarded TGDs even when T_1 is empty, and decidable for frontier-one TGDs.

1 Introduction

Tuple-generating dependencies (TGDs) are an expressive constraint language that emerged in database theory, where it has various important applications (Abiteboul, Hull, and Vianu 1995). In knowledge representation, TGDs are used as an ontology language under the names of existential rules and Datalog$^\pm$ (Baget et al. 2011; Calì et al. 2010). For the purposes of this paper, however, we stick with the name of 'TGDs'. A major application of TGDs in KR is ontology-mediated querying where a database query is enriched with an ontology, aiming to deliver more complete answers and to extend the vocabulary available for query formulation (Bienvenu et al. 2014; Bienvenu and Ortiz 2015; Calvanese et al. 2009). The semantics of ontology-mediated querying can be given in terms of homomorphisms and the widely known chase procedure that makes explicit the logical consequences of a set of TGDs and a database.

As the use of unrestricted TGDs makes the evaluation of ontology-mediated queries undecidable, various computationally more well-behaved fragments have been identified. We consider linear TGDs, guarded TGDs, and frontier-one TGDs (Calì, Gottlob, and Lukasiewicz 2012; Baget et al. 2011; Calì, Gottlob, and Kifer 2013). For all of these, ontology-mediated query evaluation is decidable. Deferring a formal definition to Section 2 of this paper, we remark that guarded generalizes linear, and that frontier-one is orthogonal to both linear and guarded. Moreover, linear TGDs generalize description logics (DLs) of the DL-Lite family (Artale et al. 2009), while both guarded and frontier-one TGDs generalize DLs of the $\mathcal{ELI}$ family (Baader et al. 2017).

On top of bare-bones query evaluation, there are other natural problems that are suggested by the framework of ontology-mediated querying. Consider the following: given sets of TGDs T_1 and T_2 (formulated in any, potentially different schemas), a database schema Σ_D, and a query schema Σ_Q, decide whether T_2 is a Σ_D, Σ_Q-*CQ-conservative extension* of T_1, that is, whether for all Σ_D-databases D and conjunctive queries (CQ) $q(\bar{x})$ in schema Σ_Q, every tuple $\bar{c}$ that is an answer to q on D given T_1 is also an answer to q on D given T_2 (Botoeva et al. 2016). Note that this is a very relevant problem. If, for instance, T_2 is a Σ_D, Σ_Q-CQ-conservative extension of T_1 and vice versa, then we can safely replace T_1 with T_2 in any application where databases are formulated in schema Σ_D and queries in schema Σ_Q. CQ-conservative extensions have been studied for various DLs and are decidable for many members of the DL-Lite and $\mathcal{ELI}$ families (Konev et al. 2011; Jung et al. 2020). In this paper, we address the naturally emerging question whether decidability extends to the more general settings of linear, guarded, and frontier-one TGDs.

A natural problem related to CQ-conservative extensions is Σ_D, Σ_Q-*hom-conservative extension* which asks whether for every Σ_D-database, there is a Σ_Q-homomorphism[1] from the chase $\mathsf{chase}_{T_2}(D)$ of D with T_2 to $\mathsf{chase}_{T_1}(D)$ that is the identity on all constants in D. In fact, this problem corresponds to CQ-conservative extensions when CQs may be infinitary, and it is known that these two problems do not coincide even in the case of DLs (Botoeva et al. 2016). We study hom-conservative extensions along with CQ-conservative extensions. In addition, we consider the variant of CQ/hom-conservative extensions where the set of TGDs T_1 is required to be empty. We refer to this as Σ_D, Σ_Q-*CQ/hom-triviality*. Note that triviality is also a very natural problem as it asks whether the given set of TGDs T_2 says *anything at all* about Σ_D-databases as far as conjunctive queries and homomorphisms over schema Σ_Q are concerned. We observe that Σ_D, Σ_Q-CQ-triviality and Σ_D, Σ_Q-hom-triviality coincide even for unrestricted TGDs, and thus we only speak of Σ_D, Σ_Q-*triviality*. Our main results are as follows.

1. For linear TGDs, CQ- and hom-conservative extensions are undecidable, but triviality is decidable.
2. For guarded TGDs, triviality is undecidable.
3. For frontier-one TGDs, CQ- and hom-conservative extensions are decidable.

[1]A homomorphism that disregards symbols outside of Σ_Q.

We consider it remarkable that undecidability already appears for a class as restricted as linear TGDs. Regarding Point 1, we also determine the exact complexity of triviality for linear TGDs as being PSPACE-complete, and CONP-complete when the arity of relation symbols is bounded by a constant. Regarding Point 3, our algorithms yield 3EXPTIME upper bounds, while 2EXPTIME lower bounds can be imported from the DL $\mathcal{ELI}$, a fragment of frontier-one TGDs (Gutiérrez-Basulto, Jung, and Sabellek 2018; Jung et al. 2020). The exact complexity remains open.

Our undecidability results are proved by reductions from a convergence problem that concerns Conway functions (Conway 1972). In a database theory context, such a technique has been used in (Gogacz and Marcinkowski 2014). As the reader shall see, the reductions take place in the setting of Pyramus and Thisbe (Ovid 2008), a mythological couple that could only communicate through a crack in the wall and whose fate it was to never meet again in person. Bring some popcorn. The decidability result for hom-conservative extensions for frontier-one TGDs rests on the observation that whenever there is a database that witnesses non-conservativity, then there is such a database of bounded treewidth. This enables a decision procedure based on alternating tree automata. The case of CQ-conservative extensions is more intricate as it requires the use of *homomorphism limits*, that is, families of homomorphisms that can only look n steps 'into the model', for any n. It is not clear how the existence of homomorphism limits can be verified by tree automata. Our solution generalizes the approach to CQ-conservative extensions in $\mathcal{ELI}$ pursued in (Jung et al. 2020). In short, the idea is to push the use of homomorphism limits to parts of the chase that are Σ_Q-disconnected from the database and regular in shape, and to then characterize homomorphism limits from/into such regular (infinite) databases in terms of unbounded homomorphisms.

Related Work. We already mentioned the work on DLs from the DL-Lite and $\mathcal{ELI}$ families (Konev et al. 2011; Jung et al. 2020). For description logics such as $\mathcal{ALC}$ that support negation and disjunction, CQ- and hom-conservative extensions are undecidable (Botoeva et al. 2019). A different kind of conservative extension is obtained by replacing databases and query answers with logical consequences formulated in the ontology language (Ghilardi, Lutz, and Wolter 2006). While such conservative extensions are decidable in $\mathcal{ALC}$ (Ghilardi, Lutz, and Wolter 2006; Lutz, Walther, and Wolter 2007), they are undecidable in the guarded fragment and in the two-variable fragment of first-order logic (Jung et al. 2017). For existential rule languages, the difference between this version of conservative extensions and CQ-conservative extensions tends to be small (depending on the class of rules considered).

2 Preliminaries

Relational Databases. Fix countably infinite and pairwise disjoint sets of *constants* $\mathbf{C}$ and $\mathbf{N}$ and variables $\mathbf{V}$. We refer to the constants in $\mathbf{N}$ as *nulls*. A *schema* Σ is a set of relation symbols R with associated arity $\mathsf{ar}(R) \geq 1$. A Σ-*fact* is an expression of the form $R(\bar{c})$ with $R \in \Sigma$ and $\bar{c}$ is an $\mathsf{ar}(R)$-tuple of constants from $\mathbf{C} \cup \mathbf{N}$. A Σ-*instance* is a possibly infinite set of Σ-facts, and a Σ-*database* is a finite Σ-instance that uses only constants from $\mathbf{C}$. We write $\mathsf{adom}(I)$ for the set of constants from $\mathbf{C} \cup \mathbf{N}$ used in instance I. For an instance I and a schema Σ, $I|_\Sigma$ denotes the restriction of I to Σ, that is, the set of all facts in I that use a relation symbol from Σ. We say that I is *connected* (resp., Σ-*connected*) if the Gaifman graph of I (resp., $I|_\Sigma$) is connected and that I is of *finite degree* if the Gaifman graph of I has finite degree.

For a schema Σ, a Σ-*homomorphism* from instance I to instance J is a function $h : \mathsf{adom}(I) \to \mathsf{adom}(J)$ such that $R(h(\bar{c})) \in J$ for every $R(\bar{c}) \in I$ with $R \in \Sigma$. We say that h is *database-preserving* if it is the identity on all constants from $\mathbf{C}$ (but not necessarily from $\mathbf{N}$) and write $I \to_\Sigma J$ if there is a database-preserving Σ-homomorphism from I to J.

Conjunctive Queries. A *conjunctive query* (CQ) over a schema Σ takes the form $\exists \bar{y}\, \phi(\bar{x}, \bar{y})$ where $\bar{x}$ and $\bar{y}$ are tuples of variables from $\mathbf{V}$, ϕ is a set of *atoms* $R(\bar{z})$ with $R \in \Sigma$ and $\bar{z}$ a tuple of variables of length $\mathsf{ar}(R)$. We refer to the variables in $\bar{x}$ as the *answer variables* of q and denote a CQ with $q(\bar{x})$ to emphasize that it has answer variables $\bar{x}$. The *arity* of q is the length $|\bar{x}|$ of $\bar{x}$, and q is *Boolean* if it is of arity 0.

Every CQ $q(\bar{x})$ gives rise to a database D_q, known as the *canonical database* of q, by viewing variables as constants and atoms as facts. A Σ-*homomorphism* h from q to an instance I is a Σ-homomorphism from D_q to I. A tuple $\bar{c} \in \mathsf{adom}(I)^{|\bar{x}|}$ is an *answer* to q on I if there is a homomorphism h from q to I with $h(\bar{x}) = \bar{c}$. The *evaluation of* $q(\bar{x})$ *on* I, denoted $q(I)$, is the set of all answers to q on I.

For a CQ q, but also for any other syntactic object q, we use $||q||$ to denote the number of symbols needed to write q encoded as a word over a suitable alphabet.

TGDs. A *tuple-generating dependency* (TGD) ϑ is a first-order sentence $\forall \bar{x} \forall \bar{y} \left(\phi(\bar{x}, \bar{y}) \to \exists \bar{z}\, \psi(\bar{x}, \bar{z}) \right)$ such that $q_\varphi = \exists \bar{y}\, \phi(\bar{x}, \bar{y})$ and $q_\psi = \exists \bar{z}\, \psi(\bar{x}, \bar{z})$ are CQs. We call ϕ and ψ the *body* and *head* of ϑ. The body may be the empty conjunction, that is, logical truth. The variables in $\bar{x}$ are the *frontier variables*. We may write ϑ as $\phi(\bar{x}, \bar{y}) \to \exists \bar{z}\, \psi(\bar{x}, \bar{z})$. An instance I *satisfies* ϑ, denoted $I \models \vartheta$, if $q_\phi(I) \subseteq q_\psi(I)$. It *satisfies* a set of TGDs T if $I \models \vartheta$ for each $\vartheta \in T$. We then also say that I is a *model* of T.

A TGD ϑ is *frontier-one* if it has exactly one frontier variable (Baget et al. 2011). It is *guarded* if its body is empty or contains a *guard atom* α that contains all variables in the body (Calì, Gottlob, and Kifer 2013). A TGD is *linear* if its body contains at most one atom. Clearly, every linear TGD is guarded. The *body width* of a set T of TGDs is the maximum number of variables in a rule body of a TGD in T, and the *head width* is defined accordingly.

Throughout this paper, we are going to make use of the well-known chase procedure for making explicit the consequences of a set of TGDs (Johnson and Klug 1984; Fagin et al. 2005; Calì, Gottlob, and Kifer 2013). Let I be an instance and T a set of TGDs. A TGD $\phi(\bar{x}, \bar{y}) \to \exists \bar{z}\, \psi(\bar{x}, \bar{z}) \in T$ is *applicable* at a tuple $\bar{c}$ of constants in I if $\phi(\bar{c}, \bar{c}') \subseteq I$ for some $\bar{c}'$ and there is no homomorphism h from $\psi(\bar{x}, \bar{z})$ to I such that $h(\bar{x}) = \bar{c}$. In this case, the *result* *of applying the TGD in I at $\bar{c}$* is the instance $I \cup \{\psi(\bar{c}, \bar{c}'')\}$ where $\bar{c}''$ is the tuple obtained from $\bar{z}$ by replacing each variable z with a fresh null, that is, a null that does not occur in I. We also refer to such an application as a *chase step*.

A *chase sequence for I with T* is a sequence of instances $I_0, I_1, \dots$ such that $I_0 = I$ and each I_{i+1} is the result of a chase step from I_i. The *result* of the chase sequence is the instance $J = \bigcup_{i \geq 0} I_i$. The chase sequence is *fair* if whenever a TGD from T is applicable to a tuple $\bar{c}$ in some I_i, then this application is a chase step in the sequence. Every fair chase sequence for I with T has the same result, up to homomorphic equivalence. Since for our purposes all results are equally useful, we use $\mathsf{chase}_T(I)$ to denote the result of an arbitrary, but fixed chase sequence for I with T and call $\mathsf{chase}_T(I)$ the *result of chasing I with T*. This version of the chase is often called the *restricted chase* and it ensures that $\mathsf{chase}_T(D)$ has finite degree, which shall be important for our proofs.

Lemma 1. *Let T be a set of TGDs and I an instance. Then for every model J of T with $I \subseteq J$, there is a homomorphism h from $\mathsf{chase}_T(I)$ to J that is the identity on $\mathsf{adom}(I)$.*

Note that if T is a set of frontier-one TGDs, then for any database D the instance $\mathsf{chase}_T(D)$ can be obtained from D by 'glueing' a (potentially infinite) instance onto each constant $c \in \mathsf{adom}(D)$. We denote this instance with $\mathsf{chase}_T(D)|^{\downarrow}_c$. A precise definition is given in the appendix.

Let T be a set of TGDs, $q(\bar{x})$ a CQ and D a database. A tuple $\bar{c} \in \mathsf{adom}(D)^{|\bar{x}|}$ is an *answer* to q on D w.r.t. T, written $D, T \models q(\bar{c})$, if $q(\bar{c})$ is logically follows from $D \cup T$ or, equivalently, if there is a homomorphism h from q to $\mathsf{chase}_T(D)$ with $h(\bar{x}) = \bar{c}$. The *evaluation of q on D w.r.t. T*, denoted $q_T(D)$, is the set of all answers to q on D w.r.t. T.

3 Conservative Extensions

We introduce the notions of conservative extension that are studied in this paper and the associated decision problems.

Definition 1. *Let T_1, T_2 be sets of TGDs and let Σ_D, Σ_Q be schemas called the* data schema *and* query schema*. Then*

- T_2 is Σ_D, Σ_Q-hom-conservative over T_1, *written $T_1 \models^{hom}_{\Sigma_D, \Sigma_Q} T_2$, if there is a database-preserving Σ_Q-homomorphism from $\mathsf{chase}_{T_2}(D)$ to $\mathsf{chase}_{T_1}(D)$ for all Σ_D-databases D;*
- T_2 is Σ_D, Σ_Q-CQ-conservative over T_1, *written $T_1 \models^{CQ}_{\Sigma_D, \Sigma_Q} T_2$, if $q_{T_2}(D) \subseteq q_{T_1}(D)$ for all Σ_D-databases D and all CQs q over schema Σ_Q.*
- T_1 *is* Σ_D, Σ_Q-hom-trivial *if T_1 is Σ_D, Σ_Q-hom-conservative over the empty set of TGDs, and likewise for* Σ_D, Σ_Q-CQ-triviality.

It is easy to see that logical entailment $T_1 \models T_2$ implies $T_1 \models^{\text{hom}}_{\Sigma_D, \Sigma_Q} T_2$ for all schemas Σ_D and Σ_Q, and that Σ_D, Σ_Q-hom-conservativity implies Σ_D, Σ_Q-CQ-conservativity. The following example from (Botoeva et al. 2016) shows that the converse fails.

Example 1. *Consider the following sets of TGDs that are both linear and frontier-one:*

$$\begin{aligned} T_1 \;=\; \{\; & A(x) \rightarrow \exists y\, S(x,y), B(y), \\ & B(x) \rightarrow \exists y\, R(x,y), B(y) \;\} \\ T_2 \;=\; \{\; & A(x) \rightarrow \exists y\, S(x,y), B(y), \\ & B(x) \rightarrow \exists y\, R(y,x), B(y) \;\}. \end{aligned}$$

Let $\Sigma_D = \{A\}$ and $\Sigma_Q = \{R\}$. We recommend to the reader to verify that T_2 is not Σ_D, Σ_Q-hom-conservative over T_1 by trying to find a database-preserving homomorphism from $\mathsf{chase}_{T_2}(D)$ to $\mathsf{chase}_{T_1}(D)$, and that it is Σ_D, Σ_Q-CQ-conservative.

However, Σ_D, Σ_Q-hom-conservativity is equivalent to Σ_D, Σ_Q-CQ-conservativity with infinitary CQs. We refrain from making this precise and instead consider the converse, that is, Σ_D, Σ_Q-CQ-conservativity is equivalent to Σ_D, Σ_Q-hom-conservativity when the latter is defined in terms of a finitary version of homomorphisms that we introduce next.

Let I_1, I_2 be instances and $n \geq 0$, and let Σ be a schema. We write $I_1 \rightarrow^n_\Sigma I_2$ if for every induced subinstance I of I_1 with $|\mathsf{adom}(I)| \leq n$, there is a database-preserving Σ-homomorphism from I to I_2. We further write $I_1 \rightarrow^{\mathsf{lim}}_\Sigma I_2$ if $I_1 \rightarrow^n_\Sigma I_2$ for all $n \geq 1$.

Theorem 1. *Let T_1 and T_2 be sets of TGDs and Σ_D, Σ_Q schemas. Then $T_1 \models^{CQ}_{\Sigma_D, \Sigma_Q} T_2$ iff $\mathsf{chase}_{T_2}(D) \rightarrow^{\mathsf{lim}}_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$.*

For triviality, the hom- and CQ-version coincide.

Lemma 2. *Let T_1, T_2 be sets of TGDs and Σ_D, Σ_Q schemas. Then T_1 and T_2 are Σ_D, Σ_Q-hom-trivial if and only if they are Σ_D, Σ_Q-CQ-trivial.*

Because of Lemma 2, we from now on disregard Σ_D, Σ_Q-CQ-triviality and refer to Σ_D, Σ_Q-hom-triviality simply as Σ_D, Σ_Q-*triviality*. We thus obtain the three decision problems *hom-conservativity*, *CQ-conservativity*, and *triviality*, defined in the obvious way. For instance, hom-conservativity means to decide, given finite sets of TGDs T_1, T_2 and finite schemas Σ_D, Σ_Q, whether T_2 is Σ_D, Σ_Q-hom-conservative over T_1.

We note that Lemma 2 is an immediate consequence of Theorem 1 and the following observation.

Lemma 3. *Let I_1, I_2 be instances such that I_1 is countable and I_2 is finite, and let Σ be a schema. If $I_1 \rightarrow^{\mathsf{lim}}_\Sigma I_2$, then $I_1 \rightarrow_\Sigma I_2$.*

We sketch the proof of Lemma 3, details are in the appendix. If $I_1 \rightarrow^{\mathsf{lim}}_\Sigma I_2$, then we find database-preserving Σ-homomorphisms $h_1, h_2, \dots$ from finite subinstances $J_1 \subseteq J_2 \subseteq \dots$ of I_1 to I_2 such that $I_1 = \bigcup_{i \geq 1} J_i$. If $h_1, h_2, \dots$ are compatible in the sense that $h_i(c) = h_j(c)$ whenever $h_i(c), h_j(c)$ are both defined, then $\bigcup_{i \geq 1} h_i$ is a Σ-homomorphism that witnesses $I_1 \rightarrow_\Sigma I_2$. If this is not the case, however, we can still manipulate $h_1, h_2, \dots$ into a compatible sequence $g_1, g_2, \dots$ by 'skipping homomorphisms', which is used in several proofs in this paper. We start with h_1 and observe that since J_1 and I_2 are finite, there are only

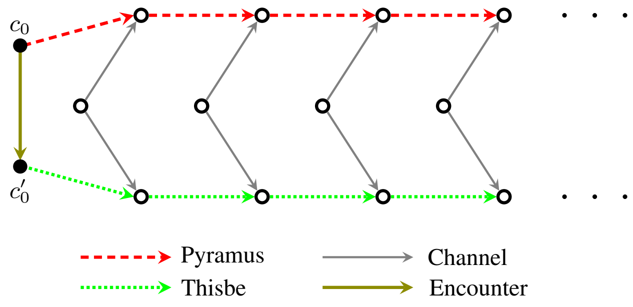

Figure 1: Chase generated by T_{myth}.

finitely many homomorphisms h from J_1 to I_2. Some such homomorphism must occur infinitely often in the restrictions of $h_1, h_2, \ldots$ to $\mathsf{adom}(J_1)$ and thus we find a subsequence $h_1', h_2', \ldots$ of $h_1, h_2, \ldots$ in which h_1' is compatible with all of $h_2', h_3', \ldots$. We proceed in the same way for h_2', then for h_3', ad infinitum, finding the desired sequence $g_1, g_2, \ldots$.

4 Undecidability

The aim of this section is to prove the following results.

Theorem 2. *The following problems are undecidable:*

1. *hom-conservativity for linear TGDs;*
2. *CQ-conservativity for linear TGDs;*
3. *triviality for guarded TGDs.*

We give a single proof that establishes Points 1 and 2. Attaining Point 3 requires a non-trivial modification of the proof. We start with the former, first highlighting the main mechanism that we use in our reduction.

4.1 The Main Mechanism

Consider the set of rules T_{myth}. It comprises three TGDs:

$$\begin{array}{rcl}
\mathsf{Encounter}(p,t) & \to & \exists p', c, t'\ M(p,p',c,t',t) \\
M(p,p',c,t',t) & \to & \exists p'', c', t''\ M(p',p'',c',t'',t') \\
M(p,p',c,t',t) & \to & \mathsf{Pyramus}(p,p'), \mathsf{Thisbe}(t,t'), \\
 & & \mathsf{Channel}(c,p'), \mathsf{Channel}(c,t').
\end{array}$$

Now consider the database $D = \{\mathsf{Encounter}(c_0, c_0')\}$. The instance $\mathsf{chase}_{T_{\mathsf{myth}}}(D)$, shown in Figure 1, will play an important role. Its intuitive meaning is that '*after an initial brief encounter, Pyramus and Thisbe have never met again, but forever remained able to connect via an (indirect) channel*.' Notice that we do not explicitly show relation M in Figure 1 as M is only a construction aid, needed to ensure that the TGDs in T_{myth} are linear. As Σ_Q, we will use the set of relation symbols in T_{myth} except M, plus a unary relation symbol Mouth. We advise the reader to not worry about the schema Σ_D at this point (it will actually be empty).

Let $\kappa = \langle [p_1, \ldots, p_n], [t_1, \ldots, t_n] \rangle$ be a pair of sequences of positive integers of the same length n. By River_κ, we mean the database that contains the following facts, an example being displayed in Figure 2:

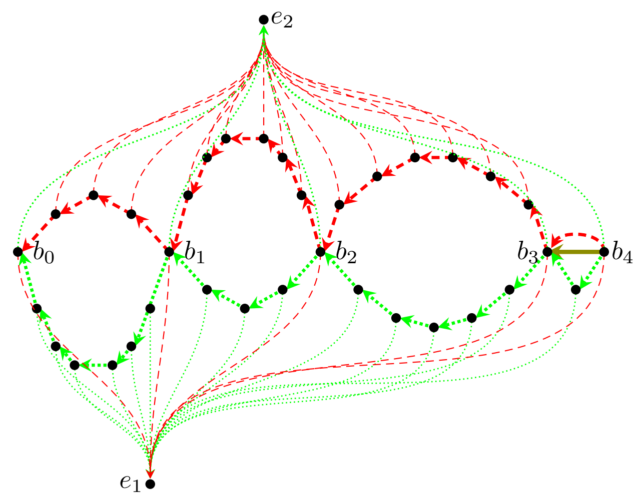

Figure 2: The database River_κ for $\kappa = \langle [4,7,7,1], [7,4,6,2] \rangle$.

- There are 3 kinds of constants. The *eternities* are e_1 and e_2. The *channel* is c, not shown in the picture. All remaining constants are called *worldly*.
- For $1 \le i \le n$ there is a Pyramus path of length p_i from b_i to b_{i-1} as well as a Thisbe path of length t_i from b_i to b_{i-1}. Constants b_i are called *bridges*.
- There is $\mathsf{Thisbe}(a, e_1)$ for each non-bridge constant a on each of the Thisbe-paths and there is $\mathsf{Pyramus}(a, e_2)$ for each non-bridge constant a on each of the Pyramus-paths. There are also $\mathsf{Thisbe}(a, e_2)$ and $\mathsf{Pyramus}(a, e_1)$ for each bridge constant a. In addition (and not in Figure 2), there are $\mathsf{Pyramus}(e_i, e_i)$ and $\mathsf{Thisbe}(e_i, e_i)$ for $i \in \{1, 2\}$.
- For each *worldly* constant a, there is $\mathsf{Channel}(c, a)$. Moreover, there are facts $\mathsf{Channel}(e_i, e_i)$, for $i \in \{1, 2\}$. These facts are not shown in Figure 2.
- There are $\mathsf{Encounter}(b_n, b_{n-1})$ and $\mathsf{Mouth}(b_0)$.

It is easy to see that $\mathsf{chase}_{T_{\mathsf{myth}}}(\mathsf{River}_\kappa)$ is obtained from River_κ by adding a copy of the instance shown in Figure 1, glueing the Encounter fact to the Encounter fact in River_κ (and adding some M-facts that are not important here). Now, let us leave to our readers the pleasure to notice that:

Observation 1. *There is a database-preserving Σ_Q-homomorphism from* $\mathsf{chase}_{T_{\mathsf{myth}}}(\mathsf{River}_\kappa)$ *to* River_κ *if and only if there exists $m \in \{1, \ldots, n-1\}$ such that $t_m \neq p_{m+1}$.*

Hint: As long as Pyramus and Thisbe walk down their respective river banks they are connected via the constant c. But for their union to last forever they need, at some point, to enter one of the eternities. Since eternity has no channel with the worldly constants (and the two eternities are not connected by a channel either), Pyramus and Thisbe both need to enter the same eternity, and they need to do it simultaneously. But this can only happen when one of them is in a bridge constant and the other in a non-bridge. □

That's nice, isn't it? But where could undecidability be lurking here?

4.2 Conway Functions

Let $\gamma, \alpha_0, \beta_0, \ldots, \alpha_{\gamma-1}, \beta_{\gamma-1}$ be positive integers such that $\beta_k | \gamma$ and $\beta_k | k\alpha_k$ for $0 \leq k < \gamma$ where as usual '|' denotes divisibility without remainder. For a positive integer n, define $\digamma(n)$ by setting $\digamma(n) = n\alpha_k/\beta_k$ for $k = n \bmod \gamma$. Thus, the remainder of n when dividing by γ determines the pair (α_k, β_k) used to compute the value $\digamma(n)$. Note that due to the two divisibility conditions, the range of $\digamma$ contains only positive integers.

The function $\digamma$ is called the *Conway function defined by* $\gamma, \alpha_0, \beta_0, \ldots, \alpha_{\gamma-1}, \beta_{\gamma-1}$. We say that $\digamma$ *stops* if there exists an $n \in \mathbb{N}$ such that $\digamma^n(2) = 1$, where $\digamma^n$ is $\digamma$ composed with itself, n times. There is no special meaning to the numbers 1 and 2 used here, we could choose otherwise. The following is well-known, see also (Gogacz and Marcinkowski 2014).

Theorem 3. *It is undecidable whether the Conway function defined by a given sequence* $\gamma, \alpha_0, \beta_0, \ldots, \alpha_{\gamma-1}, \beta_{\gamma-1}$ *stops.*

Take a sequence $\gamma, \alpha_0, \beta_0, \ldots, \alpha_{\gamma-1}, \beta_{\gamma-1}$ defining a Conway function $\digamma$. We prove Points 1 and 2 of Theorem 2 by showing how to compute, given the sequence, sets T_1 and T_2 of linear TGDs along with schemas Σ_D, Σ_Q such that $\digamma$ does not stop if and only if T_2 is Σ_D, Σ_Q-hom-conservative over T_1 if and only if T_2 is Σ_D, Σ_Q-CQ-conservative over T_1. We assume without loss of generality that $\digamma(1) = 1$ and $\digamma(2) = 3$.

4.3 The Reduction

We say that $\kappa = \langle [p_1, \ldots, p_n], [t_1, \ldots, t_n] \rangle$ (or River_κ) is

- *locally correct* if the following conditions hold:
  1. $p_1 = 2$ and $p_n = 1$;
  2. $\digamma(p_i) = t_i$ for $1 \leq i < n$;
- *correct* if it is locally correct and $t_i = p_{i+1}$ for $1 \leq i < n$.

The database River_κ shown in Figure 2 is not locally correct because $p_1 \neq 2$ and $t_n \neq 1$ (which must be the case as we assume $\digamma(1) = 1$).

Clearly, $\digamma$ does not stop if and only if every locally correct River_κ is incorrect, and by Observation 1 this is the case if and only if for each locally correct sequence κ there exists a database-preserving Σ_Q-homomorphism from $\mathsf{chase}_{T_{\mathsf{myth}}}(\mathsf{River}_\kappa)$ to River_κ.

Now the plan is as follows. Take $\Sigma_D = \emptyset$. We define T_1 such that $\mathsf{chase}_{T_1}(\emptyset)$ is the 'disjoint union' of all locally correct databases River_κ. Our T_2 will be the union of T_1 and T_{myth}. A careful reader can notice that if this plan succeeds, then the proof of Point 1 of Theorem 2 will be completed. And it will indeed succeed, but not without one little nuance. This is the reason why we used quotations mark around the term 'disjoint union' above.

The set of TGDs T_1 is the union of two sets of linear TGDs T_{rec} and T_{proj}. As intended, T_1 generates the union of all locally correct databases River_κ. The mentioned nuance is that the union is not disjoint, but massively overlapping. However, this does not compromise correctness of the reduction.

The rules of T_{rec} will not mention symbols from Σ_Q. They instead use a schema $\Sigma_\digamma$ that consists of high arity relation symbols used as construction aids. We later use T_{proj} to relate these symbols to those in Σ_Q. More precisely, $\Sigma_\digamma$ contains relation symbols Start of arity 8, End of arity 5, Bridge of arity 4, WH^i_k (for WorkHorse) of arity $\alpha_k + \beta_k + 5$ for $0 \leq k, i < \gamma$, and BH_k (for BridgeHead) of arity $\alpha_k + \beta_k + 5$ for $0 \leq k < \gamma$. In what follows, we use $\dagger$ to denote the list of variables 'c, e_1, e_2'. With $+_\gamma$ and $-_\gamma$, we denote addition and subtraction in the ring $\mathbb{Z}_\gamma$.

Since $\Sigma_D = \emptyset$, first of all we need a rule that will create something out of nothing:

$$\rightarrow \quad \exists \dagger, b_0, x_1, y_1, y_2, b_1 \; \mathsf{Start}(\dagger, b_0, x_1, y_1, y_2, b_1).$$

Later, T_{proj} will generate a Pyramus-path from b_1 via x_1 to b_0 and a Thisbe-path from b_1 via y_2 and y_1 to b_0, determining the lengths $p_1 = 2$ and $t_1 = 3$ of the river. Recall that local correctness prescribes $p_1 = 2$ and we assume $\digamma(2) = 3$. We need to know that b_1 is a bridge:

$$\mathsf{Start}(\dagger, b_0, x_1, y_1, y_2, b_1) \;\rightarrow\; \mathsf{Bridge}(\dagger, b_1).$$

We now put our horses to work by adding, for $0 \leq k < \gamma$:

$$\mathsf{Bridge}(\dagger, b) \rightarrow \exists x_1, \ldots, x_{\beta_i}, y_1, \ldots, y_{\alpha_i} \; \mathsf{WH}^{\beta_k}_k(\dagger, b, x_1, \ldots, x_{\beta_k}, b, y_1, \ldots, y_{\alpha_k})$$

and for $0 \leq k, i < \gamma$:

$$\mathsf{WH}^i_k(\dagger, x_0, x_1, \ldots, x_{\beta_k}, y_0, y_1, \ldots, y_{\alpha_k}) \rightarrow \exists z_1, \ldots, z_{\beta_k}, u_1, \ldots, u_{\alpha_k} \; \mathsf{WH}^{i +_\gamma \beta_k}_k(\dagger, x_{\beta_k}, z_1, \ldots, z_{\beta_k}, y_{\alpha_k}, u_1, \ldots, u_{\alpha_k}).$$

$\mathsf{WH}^i_k(\dagger, c_0, c_1, \ldots, x_{\beta_k}, y_0, y_1, \ldots, y_{\alpha_k})$ promises to generate, via T_{proj}, a Pyramus-path of length β_k from x_{β_k} to x_0 and a Thisbe-path of length α_k from y_{α_k} to y_0. The above two rules thus patiently produce Pyramus- and Thisbe-paths that lead to b. The superscript $\cdot^i$ remembers how many Pyramus-edges have been produced since the last bridge, modulo γ, and the subscript $\cdot_k$ chooses a remainder class, that is, it expresses the promise that the Pyramus-path between the two bridges is of length n, for some number n with $n \bmod \gamma = k$.

Then, at some point, the next bridge can be reached:

$$\mathsf{WH}^{k -_\gamma \beta_k}_k(\dagger, x_0, x_1, \ldots, x_{\beta_k}, y_0, y_1, \ldots, y_{\alpha_k}) \rightarrow \exists z_1, \ldots, z_{\beta_k - 1}, u_1, \ldots, u_{\alpha_k - 1}, b \; \mathsf{BH}_k(\dagger, x_{\beta_k}, z_1, \ldots, z_{\beta_k - 1}, b, y_{\alpha_k}, u_1, \ldots, u_{\alpha_k - 1}, b)$$

$$\mathsf{BH}_k(\dagger, x_{\beta_k}, z_1, \ldots, z_{\beta_k - 1}, b, y_{\alpha_k}, u_1, \ldots, u_{\alpha_k - 1}, b) \rightarrow \mathsf{Bridge}(\dagger, b).$$

In the first rule above, relation $\mathsf{WH}^{k -_\gamma \beta_k}_k$ indicates that we have seen m Pyramus-edges, for some m with $m \bmod \gamma = k -_\gamma \beta_k$, and that BH_k will generate β_k more Pyramus-edges, thus arriving at the promised remainder of k. It is also easy to see that if the chosen remainder class was k and the length of the Pyramus-path between two bridges produced by the above rules is n, then the length of the Thisbe-path is $\digamma(n) = n\alpha_k/\beta_k$. Thus, Point 2 of local correctness is satisfied.

Finally, we want to produce the last[2] segment of the river:

$$\mathsf{Bridge}(\dagger, b) \rightarrow \exists b' \; \mathsf{End}(\dagger, b, b')$$

[2]Orographically the first, as we generate the river from the mouth to the source.

This will generate direct Pyramus- and Thisbe-edges from b' to b (recall that $F(1) = 1$).

The TGDs in T_{rec} generate the actual rivers as projections of the template produced by T_{rec}. We start at the mouth:

$$\begin{aligned}&\mathsf{Start}(\dagger, b_0, x_1, y_1, y_2, b_1) \to\\ &\quad \mathsf{Mouth}(b_0),\\ &\quad \mathsf{Pyramus}(x_1, b_0), \mathsf{Pyramus}(b_1, x_1), \mathsf{Pyramus}(x_1, e_2),\\ &\quad \mathsf{Thisbe}(y_1, b_0), \mathsf{Thisbe}(y_2, y_1), \mathsf{Thisbe}(b_1, y_2),\\ &\quad \mathsf{Thisbe}(y_1, e_1), \mathsf{Thisbe}(y_2, e_1),\\ &\quad \mathsf{Channel}(c, x_1), \mathsf{Channel}(c, y_1), \mathsf{Channel}(c, y_2),\\ &\quad \mathsf{Channel}(e_1, e_1), \mathsf{Pyramus}(e_1, e_1), \mathsf{Thisbe}(e_1, e_1)\\ &\quad \mathsf{Channel}(e_2, e_2), \mathsf{Pyramus}(e_2, e_2), \mathsf{Thisbe}(e_2, e_2).\end{aligned}$$

The rules for WH_k^i are then as expected:

$$\begin{aligned}&\mathsf{WH}_k^i(\dagger, x_0, x_1, \dots, x_{\beta_k}, y_0, y_1, \dots, y_{\alpha_k}) \to\\ &\quad \mathsf{Pyramus}(x_1, x_0), \dots, \mathsf{Pyramus}(x_{\beta_k}, x_{\beta_k - 1}),\\ &\quad \mathsf{Pyramus}(x_1, e_2), \dots, \mathsf{Pyramus}(x_{\beta_k}, e_2),\\ &\quad \mathsf{Thisbe}(y_1, y_0), \dots, \mathsf{Thisbe}(y_{\alpha_k}, y_{\alpha_k - 1}),\\ &\quad \mathsf{Thisbe}(y_1, e_1), \dots, \mathsf{Thisbe}(y_{\alpha_k}, e_1),\\ &\quad \mathsf{Channel}(c, x_1), \dots, \mathsf{Channel}(c, x_{\beta_k}),\\ &\quad \mathsf{Channel}(c, y_1), \dots, \mathsf{Channel}(c, y_{\alpha_k}).\end{aligned}$$

Rules for the relations BH_i are analogous, so we skip them. There are also rules for projecting relations Bridge and End:

$$\begin{aligned}&\mathsf{Bridge}(\dagger, b) \to \mathsf{Channel}(c, b), \mathsf{Pyramus}(b, e_1), \mathsf{Thisbe}(b, e_2)\\ &\mathsf{End}(\dagger, b, b') \to \mathsf{Pyramus}(b', b), \mathsf{Thisbe}(b', b),\\ &\qquad\qquad \mathsf{Encounter}(b, b').\end{aligned}$$

In the appendix, we show that:

Lemma 4. *F does not stop iff $T_2 = T_1 \cup T_{\mathsf{myth}}$ is Σ_Q, Σ_D-hom-conservative over $T_1 = T_{\mathsf{rec}} \cup T_{\mathsf{proj}}$.*

This establishes Point 1 of Theorem 2. For the "if" direction, one shows that if $\mathsf{chase}_{T_2}(\emptyset) \to_{\Sigma_Q} \mathsf{chase}_{T_1}(\emptyset)$, then every locally correct river is incorrect, and thus F stops. Since rivers may be long, but are finite, it actually suffices that $\mathsf{chase}_{T_2}(\emptyset) \to_{\Sigma_Q}^{\mathsf{lim}} \mathsf{chase}_{T_1}(\emptyset)$ for F to stop, which by Theorem 1 gives Point 2 of Theorem 2.

For Point 3 of Theorem 2, we again want to use the toolkit above, in particular T_{myth} and Observation 1. But the situation is a bit different now. In the above reduction, we had at our disposal T_1 which was able to produce, from nothing, all the rivers we needed. So we could afford to have $\Sigma_D = \emptyset$. Now, however, we no longer have T_1, but only T_2, and our strategy is as follows. Recall that F stops if and only if there is a locally correct River_κ that is correct, and that River_κ is correct if there is no database-preserving Σ_Q-homomorphism from $\mathsf{chase}_{T_{\mathsf{myth}}}(\mathsf{River}_\kappa)$ to River_κ. We use the database D to guess a River_κ that admits no such homomorphism. More precisely, we design T_2 so that it verifies the existence of a (single) locally correct river in D and only if successful generates a chase with T_{myth} at the Encounter fact of that river. Details are in the appendix.

5 Triviality for Linear TGDs

We show that for linear TGDs, Σ_D, Σ_Q-triviality is decidable and PSPACE-complete, while it is only coNP-complete when the arity of relation symbols is bounded by a constant. The upper bounds crucially rely on the observation that non-triviality is always witnessed by a *singleton database*, that is, a database that contains at most one fact. This was first noted (for CQ-conservative extensions) in the context of the decription logic DL-Lite (Konev et al. 2011).

Lemma 5. *Let T be a set of linear TGDs and Σ_D, Σ_Q schemas. Then T is Σ_D, Σ_Q-trivial iff $\mathsf{chase}_T(D) \to_{\Sigma_Q} D$ for all singleton Σ_D-databases D.*

So an important part of deciding triviality is to decide, given a set of TGDs T and a singleton database D, whether $\mathsf{chase}_T(D) \not\to_{\Sigma_Q} D$. The basis for this is the subsequent lemma.

Lemma 6. *Let T be a set of linear TGDs and D a singleton database. Then $\mathsf{chase}_T(D) \not\to_{\Sigma_Q} D$ implies that there is a connected database $C \subseteq \mathsf{chase}_T(D)$ that contains at most two facts and such that $C \not\to_{\Sigma_Q} D$.*

Lemmas 5 and 6 provide us with a decision procedure for triviality for linear TGDs. Given a finite set of linear TGDs T and finite schemas Σ_D and Σ_Q, all we have to do is iterate over all singleton Σ_D-databases D and over all $C \subseteq \mathsf{chase}_T(D)$ that contain at most two facts and check (in polynomial time) whether $C \to_{\Sigma_Q} D$. To identify the sets C, we can iterate over all exponentially many candidates and check for each of them whether $D, T \models q_C$, where q_C is C viewed as a Boolean CQ. This entailment check is possible in PSPACE (Gottlob, Manna, and Pieris 2015). This yields the PSPACE upper bound in the following result.

Theorem 4. *For linear TGDs, triviality is* PSPACE*-complete. It is* coNP*-complete if the arity of relation symbols is bounded by a constant.*

To obtain the coNP upper bound, we recall that when the arity of relation symbols is bounded by a constant, then the entailment check '$D, T \models q_C$' is in NP (Gottlob et al. 2014). To decide non-triviality, we may thus guess D and C and verify in polynomial time that $C \not\to_{\Sigma_Q} D$ and in NP that $D, T \models q_C$. For the lower bounds, we reduce entailments of the form $D, T \models \exists x\, A(x)$, with T a set of linear TGDs, to non-triviality for linear TGDs. This problem is PSPACE-hard in general (Casanova, Fagin, and Papadimitriou 1984) and it is common knowledge that it is NP-hard when the arity of relation symbols is bounded by a constant. The reduction goes as follows. Let D, T, and $\exists x\, A(x)$ be given. Introduce a fresh binary relation symbol R, set $\Sigma_D = \Sigma_Q = \{R\}$, and let T' be the extension of T with the TGDs

$$\begin{aligned}&\to \quad q_D\\ A(u) &\to \quad \exists x \exists y \exists z\, R(x, y), R(y, z)\end{aligned}$$

where q_D is D viewed as a Boolean CQ. Note that there is no homomorphism from $R(x, y), R(y, z)$ into the singleton Σ_D-database $\{R(c, c')\}$. Based on this, it is easy to verify that T' is Σ_D, Σ_Q-trivial iff $D, T \not\models \exists x\, A(x)$.

6 Frontier-One TGDs

The purpose of this section is to show the following.

Theorem 5. *For frontier-one TGDs, CQ-conservativity and hom-conservativity are decidable in* 3ExpTime *(and* 2ExpTime*-hard).*

2ExpTime lower bounds carry over from the description logic $\mathcal{ELI}$, see (Gutiérrez-Basulto, Jung, and Sabellek 2018) for hom-conservativity and (Jung et al. 2020) for CQ-conservativity. They already apply when only unary and binary relation symbols are admitted. In the remainder of the section, we thus concentrate on the upper bounds.

Both in the case of hom-conservativity and CQ-conservativity, we first provide a suitable model-theoretic characterization and then use it to find a decision procedure based on tree automata. The case of CQ-conservativity is significantly more challenging because of the appearance of homomorphism limits.

6.1 Deciding Hom-Conservativity

We show that to decide hom-conservativity, it suffices to consider databases of bounded treewidth. Instead of using the standard notion of a tree decomposition, however, it is more convenient for us to work with so-called tree-like databases. Variations of these have been used for instance in (Benedikt, Bourhis, and Senellart 2012; Jung et al. 2018).

A Σ-*instance tree* is a triple $\mathcal{T} = (V, E, B)$ with (V, E) a directed tree and B a function that assigns a Σ-database $B(v)$ to every $v \in V$ such that the following conditions hold:

1. for every $a \in \bigcup_{v \in V} \mathsf{adom}(B(v))$, the restriction of (V, E) to the nodes $v \in V$ such that $a \in \mathsf{adom}(B(v))$ is a tree of depth at most one;
2. for every $(u, v) \in E$, $|\mathsf{adom}(B(u)) \cap \mathsf{adom}(B(v))| \leq 1$.

The *width* of the instance tree is the supremum of the cardinalities of $\mathsf{adom}(B(v))$, $v \in V$. A Σ-instance tree $\mathcal{T}$ defines an associated instance $I_\mathcal{T} = \bigcup_{v \in V} B(v)$. A Σ-instance I is *tree-like of width* k if there is a Σ-instance tree $\mathcal{T}$ of width k with $I = I_\mathcal{T}$.

Instance trees of width k are closely related to tree decompositions of width k in which the bags overlap in at most one constant. Condition 1, however, strengthens the usual connectedness condition to trees of depth 1. This strengthening is crucial for our constructions and not possible for other classes of TGDs such as guarded TGDs.

Theorem 6. *Let T_1 and T_2 be sets of frontier-one TGDs, and Σ_D and Σ_Q schemas. Let k be the body width of T_1. Then the following are equivalent:*

1. $T_1 \models^{hom}_{\Sigma_D,\Sigma_Q} T_2$;
2. $\mathsf{chase}_{T_2}(D) \rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$, *for all tree-like Σ_D-databases D of width at most k.*

The "1 $\Rightarrow$ 2"-direction is a direct consequence of the definition of hom-conservativity. For the "2 $\Rightarrow$ 1"-direction, let D be a Σ_D-database witnessing $T_1 \not\models^{\text{hom}}_{\Sigma_D,\Sigma_Q} T_2$, that is, $\mathsf{chase}_{T_2}(D) \not\rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$. We show in the appendix that the unraveling U of D into a (potentially infinite) tree-like Σ_D-instance of width k also satisfies $\mathsf{chase}_{T_2}(U) \not\rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(U)$. Compactness then yields a finite subset U' of U that still satisfies $\mathsf{chase}_{T_2}(U') \not\rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(U')$.

We show in the appendix how Theorem 6 can be used to reduce Σ_D, Σ_Q-hom-conservativity to the ExpTime-complete emptiness problem of two-way alternating tree automata (2ATAs) and in this way obtain a 3ExpTime upper bound. Here, we only give a sketch. Let T_1 and T_2 be sets of frontier-one TGDs, Σ_D and Σ_Q schemas, k the body width of T_1, and ℓ the head width of T_1.

The 2ATA works on input trees that encode a tree-like database D of width at most k along with a tree-like model I_0 of D and T_1 of width at most $\max\{k, \ell\}$. It verifies that $\mathsf{chase}_{T_2}(D) \not\rightarrow_{\Sigma_Q} I_0$. If such an I_0 is found, then $\mathsf{chase}_{T_2}(D) \not\rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$ because $\mathsf{chase}_{T_1}(D) \rightarrow I_0$. The converse is also true since $\mathsf{chase}_{T_1}(D)$ is tree-like of width $\max\{k, \ell\}$. In fact, the instance $\mathsf{chase}_{T_1}(D)|^{\downarrow}_{c}$ that the chase generates below each $c \in \mathsf{adom}(D)$ (see Section 2) is tree-like of width ℓ.

Since our homomorphisms are database-preserving and T_2 is a set of frontier-one TGDs, $\mathsf{chase}_{T_2}(D) \not\rightarrow_{\Sigma_Q} I_0$ if and only if there is a $c \in \mathsf{adom}(D)$ such that $\mathsf{chase}_{T_2}(D)|^{\downarrow}_{c} \not\rightarrow_{\Sigma_Q} I_0$. The 2ATA may thus check this latter condition, which it does by relying on the notion of a type. Since types also play a role in the subsequent sections, we make this precise.

Let T be a set of frontier-one TGDs. We use $\mathsf{bodyCQ}(T)$ to denote the set of unary or Boolean CQs that can be obtained by starting with the Boolean CQ $\exists y \exists \bar{z}\, \phi(y, \bar{z})$ with $\phi(y, \bar{z})$ the body of some TGD in T, then dropping any number of atoms, and then identifying variables. Finally, we may choose a variable as the answer variable and rename it to the fixed variable x (or stick with a Boolean CQ). A T-*type* is a subset $t \subseteq \mathsf{bodyCQ}(T)$ such that for some instance I that is a model of T and some $c \in \mathsf{adom}(I)$,

1. $q(x) \in t$ iff $c \in q(I)$ for all unary $q(x) \in \mathsf{bodyCQ}(T)$ and
2. $q \in t$ iff $I \models q$ for all Boolean $q \in \mathsf{bodyCQ}(T)$.

We then also use $\mathsf{tp}_T(I, c)$ to denote t. We assume that every type contains the additional formula $\mathsf{true}(x)$ (so that x is guaranteed to occur free in t). We may then view t as a unary CQ with free variable x and thus as a (canonical) database. For brevity, we use t also to denote both of these. $\mathsf{TP}(T)$ is the set of all T-types. Note that the number of types is double exponential in $||T||$. The type $\mathsf{tp}_T(\mathsf{chase}_T(D), c)$ tells us everything we need to know about c in the chase of a database D with T, as follows.

Lemma 7. *Let T be a set of frontier-one TGDs, I an instance, and $c \in \mathsf{adom}(I)$. Then $\mathsf{chase}_T(I)|^{\downarrow}_{c}$ and $\mathsf{chase}_T(J)|^{\downarrow}_{c}$ are homomorphically equivalent, where J is obtained from $\mathsf{tp}_T(\mathsf{chase}_T(I), c)$ by replacing the free variable x with c.*

The proof of Lemma 7 is straightforward by reproducing chase steps from the construction of $\mathsf{chase}_T(I)$ in $\mathsf{chase}_T(J)$ and vice versa. Details are omitted.

So to verify that $\mathsf{chase}_{T_2}(D) \not\rightarrow_{\Sigma_Q} I_0$, a 2ATA may guess a constant c in the database D represented by the input tree, and it may also guess the type $\mathsf{tp}_{T_2}(\mathsf{chase}_{T_2}(D), c)$. It then goes on to verify that $\mathsf{tp}_{T_2}(\mathsf{chase}_{T_2}(D), c)$ was guessed correctly (which is not entirely trivial as $\mathsf{chase}_{T_2}(D)$ is *not* encoded in the input). Exploiting Lemma 7, it then starts from type $\mathsf{tp}_{T_2}(\mathsf{chase}_{T_2}(D), c)$ to construct 'in its states' the instance $\mathsf{chase}_{T_2}(D)|^{\downarrow}_{c}$, simultaneously walking through the instance I_0 encoded by the input tree to verify that, as desired, $\mathsf{chase}_{T_2}(D)|^{\downarrow}_{c} \not\rightarrow_{\Sigma_Q} I_0$ (we actually build a 2ATA for verifying $\mathsf{chase}_{T_2}(D)|^{\downarrow}_{c} \rightarrow_{\Sigma_Q} I_0$ and then complement).

6.2 Deciding CQ-Conservativity

We start with showing that, also for deciding CQ-conservativity, it suffices to consider tree-like databases. In addition, it suffices to consider CQs q of arity 0 or 1.

Theorem 7. *Let T_1 and T_2 be sets of frontier-one TGDs, and Σ_D and Σ_Q schemas. Let k be the body width of T_1. Then the following are equivalent:*

1. $T_1 \models^{CQ}_{\Sigma_D,\Sigma_Q} T_2$;
2. $q_{T_2}(D) \subseteq q_{T_1}(D)$, *for all tree-like Σ_D-databases D of width at most k and connected Σ_Q-CQs q of arity 0 or 1.*

The proof of Theorem 7 first concentrates on restricting the shape of the database, using unraveling and compactness as in the proof of Theorem 6. In a second step, it is then not difficult to restrict also the shape of the CQ.

The following refinement of Theorem 1 is a straightforward consequence of Theorem 7.

Theorem 8. *Let T_1 and T_2 be sets of frontier-one TGDs, Σ_D and Σ_Q schemas, and k the body width of T_1. Then the following are equivalent:*

1. $T_1 \models^{CQ}_{\Sigma_D,\Sigma_Q} T_2$;
2. $\mathsf{chase}_{T_2}(D) \rightarrow^{\mathsf{lim}}_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$, *for all tree-like Σ_D-databases D of width at most k.*

Although Theorem 8 looks very similar to Theorem 6, it does not directly suggest a decision procedure. In particular, it is not clear how tree automata can deal with homomorphism limits. We next work towards a more operative characterization that pushes the use of homomorphism limits to parts of the chase that are Σ_Q-disconnected from the database and regular in shape. As we shall see, this allows us to get to grips with homomorphism limits.

For a database D, with $\mathsf{chase}_T(D)|^{\mathsf{con}}_{\Sigma}$ we denote the union of all maximally Σ-connected components of $\mathsf{chase}_T(D)$ that contain at least one constant from $\mathsf{adom}(D)$.

Theorem 9. *Let T_1 and T_2 be sets of frontier-one TGDs, Σ_D and Σ_Q schemas, and k the body width of T_1. Then $T_1 \models^{CQ}_{\Sigma_D,\Sigma_Q} T_2$ iff for all tree-like Σ_D-databases D of width at most k, the following holds:*

1. $\mathsf{chase}_{T_2}(D)|^{\mathsf{con}}_{\Sigma_Q} \rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$;
2. *for all maximally Σ_Q-connected components I of $\mathsf{chase}_{T_2}(D) \setminus \mathsf{chase}_{T_2}(D)|^{\mathsf{con}}_{\Sigma_Q}$, one of the following holds:*
   (a) $I \rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$;
   (b) $I \rightarrow^{\mathsf{lim}}_{\Sigma_Q} \mathsf{chase}_{T_1}(D)|^{\downarrow}_{c}$ *for some $c \in \mathsf{adom}(D)$.*

The subsequent example illustrates the theorem.

Example 2. *Consider the sets of TGDs T_1, T_2 and the schemas Σ_D, Σ_Q from Example 1. Recall that T_2 is Σ_D, Σ_Q-CQ-conservative over T_1. Since Σ_D contains only the unary relation A, we may w.l.o.g. concentrate on the Σ_D-database $D = \{A(c)\}$. Clearly, Point 1 of Theorem 9 is satisfied.*

For Point 2, observe that $\mathsf{chase}_{T_2}(D) \setminus \mathsf{chase}_{T_2}(D)|^{\mathsf{con}}_{\Sigma_Q}$ contains only one maximally Σ_Q-connected component, which is of the form

$$I = \{R(c_1, c_0), R(c_2, c_1), \dots\}.$$

Moreover, $I \rightarrow^{\mathsf{lim}}_{\Sigma_Q} \mathsf{chase}_{T_1}(D)|^{\downarrow}_{c}$ and thus Point 2(b) is satisfied. Point 2(a) is not satisfied since $I \not\rightarrow_{\Sigma_Q} \mathsf{chase}_{T_1}(D)$.

The easier 'if' direction of the proof of Theorem 9 relies on the fact that, as per Theorem 7, we can concentrate on connected CQs of arity 0 or 1. The interesting direction is 'only if', distinguishing several cases and using several 'skipping homomorphism' arguments (see Lemma 3).

Points 1 and 2(a) of Theorem 9 are amenable to the same tree automata techniques that we have used for hom-conservativity. Point 2(b) achieves the desired expulsion of homomorphism limits, away from the database D to instances of regular shape. In fact, the number of possible T_1-types is independent of D and thus by Lemma 7 the number of distinct instances $\mathsf{chase}_{T_1}(D)|^{\downarrow}_{c}$ in Point 2(b) that have to be considered is also independent of D. Moreover, these instances are purely chase-generated and thus regular in shape. The same is true for the instances I in Point 2. We next take a closer look at the latter.

Let T be a set of frontier-one TGDs. A *T-labeled database* is a pair $A = (D, \mu)$ with D a database and $\mu : \mathsf{adom}(D) \rightarrow \mathsf{TP}(T)$. We associate A with a database D_A that is obtained by starting with D and then adding, for each $c \in \mathsf{adom}(D)$, a disjoint copy D' of the type $\mu(c)$ viewed as a database and glueing the copy of x in D' to c in D_A. We use T-labeled databases to describe fragments of chase-generated instances, and thus assume that D_A contains only null constants. We also associate A with a Boolean CQ q_A, obtained by viewing D_A as such a CQ.

A *labeled Σ-head fragment of T_2* is a T_2-labeled database (F, μ) such that F can be obtained by choosing a TGD $\phi(x, \bar{y}) \rightarrow \exists \bar{z}\, \psi(x, \bar{z}) \in T_2$ and taking a maximally Σ-connected component of ψ that does not contain the frontier variable. The following lemma follows from an easy analysis of the chase procedure. Proof details are omitted.

Lemma 8. *Let I be a maximally Σ_Q-connected component of $\mathsf{chase}_{T_2}(D) \setminus \mathsf{chase}_{T_2}(D)|^{\mathsf{con}}_{\Sigma_Q}$, as in Point 2 of Theorem 9. Then for some labeled Σ_Q-head fragment $A = (F, \mu)$ of T_2,*

1. $\mathsf{chase}_{T_2}(D) \models q_A$, *and*
2. *I is homomorphically equivalent to $\mathsf{chase}_T(D_A)|^{\mathsf{con}}_{\Sigma_Q}$.*

Clearly, the number of labeled Σ-head fragments of T_2 is independent of D, just like the number of T_1-types. It thus follows from Lemma 8 and what was said before it that the number of checks '$I \rightarrow^{\mathsf{lim}}_{\Sigma_Q} \mathsf{chase}_{T_1}(D)|^{\downarrow}_{c}$' in Point 2(b) of Theorem 9 does not depend on D: there is at most one such check for every labeled Σ-head fragment of T_2 and every T_1-type. We can do all these checks in a preprocessing step, before starting to build 2ATAs for CQ-conservativity that implement the characterization provided by Theorem 9. Whenever the 2ATA needs to carry out a check '$I \rightarrow^{\mathsf{lim}}_{\Sigma_Q}$

$\mathsf{chase}_{T_1}(D)|^{\downarrow}_c$' to verify Point 2(b), we can simply look up the precomputed result and let the 2ATA reject immediately if it is negative. Thus, the automata are completely freed from dealing with homomorphism limits.

6.3 Precomputing Homomorphism Limits

It remains to show how to actually achieve the precomputation of the tests '$I \to^{\mathsf{lim}}_{\Sigma_Q} \mathsf{chase}_{T_1}(D)|^{\downarrow}_c$' in Point 2(b) of Theorem 9. This is where we finally deal with homomorphism limits. The following theorem makes precise the problem that we actually have to decide.

Theorem 10. *Given two sets of frontier-one TGDs T_1 and T_2, a schema Σ, a labeled Σ-head fragment $A = (D, \mu)$ for T_2, and a T_1-type $\widehat{t}$, it can be decided in time triple exponential in $||T_1|| + ||T_2||$ whether $\mathsf{chase}_{T_2}(D_A)|^{\mathsf{con}}_{\Sigma} \to^{\mathsf{lim}}_{\Sigma} \mathsf{chase}_{T_1}(\widehat{t})$.*

We invite the reader to compare the decision problem formulated in Theorem 10 with Point 2(b) of Theorem 9 in the light of Lemmas 7 and 8. The decision procedure used to prove Theorem 10 is again based on tree automata. To enable their use, however, we first rephrase the decision problem in Theorem 10 in a way that replaces homomorphism limits with unbounded homomorphisms.

Let T_1, T_2, Σ, $A = (D, \mu)$, and $\widehat{t}$ be as in Theorem 10. Recall that A is associated with a database D_A and a Boolean CQ q_A. Here, we additionally use unary CQs q^c_A, for every $c \in \mathsf{adom}(D)$, which are defined exactly like q_A except that c is now the answer variable.

The main idea for proving Theorem 10 is to replace homomorphism limits into $\mathsf{chase}_{T_1}(\widehat{t})$ with homomorphisms into a class of instances $\mathcal{R}(T_1, \widehat{t})$ whose disjoint union should be viewed as a relaxation of $\mathsf{chase}_{T_1}(\widehat{t})$. In particular, this relaxation admits a homomorphism limit to $\mathsf{chase}_{T_1}(\widehat{t})$, but not a homomorphism. Let us make this precise.

We again use instance trees. This time, however, they are not based on directed trees, but on *directed pseudo-trees*, that is, finite or infinite directed graphs $G = (V, E)$ such that every node $v \in V$ has at most one incoming edge and G is connected and contains no cycle.[3] Note that infinite directed pseudo-trees need not have a root. For example, a two-way infinite path qualifies as a directed pseudo-tree.

A T_1-*labeled instance tree* has the form $\mathcal{T} = (V, E, B, \mu)$ with $\mathcal{T}' = (V, E, B)$ an instance tree (based on a directed pseudo-tree) and $\mu : \mathsf{adom}(I_{\mathcal{T}'}) \to \mathsf{TP}(T_1)$ a function that assigns a T_1-type to every element in $I_{\mathcal{T}'}$. For $v \in V$, we use μ_v to denote the restriction of μ to $\mathsf{adom}(B(v))$. Moreover, we set $I_{\mathcal{T}} = I_{\mathcal{T}'}$. We say that $\mathcal{T}$ is $\widehat{t}$-*proper* if the following conditions are satisfied:

1. for every $v \in V$, one of the following holds:
   (a) v is the root of (V, E), $B(v)$ has the form $\{\mathsf{true}(c_0)\}$, and $\mu(c_0) = \widehat{t}$;
   (b) there is a TGD ϑ in T_1 such that $B(v)$ is isomorphic to the head of ϑ and $\widehat{t}, T_1 \models q_{(B(v),\mu_v)}$;
2. for every $(u, v) \in E$ such that $B(u) \cap B(v)$ contains a (single) constant c, we have $\mu_u(c), T_1 \models q^c_{(B(v),\mu_v)}(x)$. That is: the constant x from the type $\mu_u(c)$ viewed as a database is an answer to the unary CQ $q^c_{(B(v),\mu_v)}$ w.r.t. T_1.

[3] Neither in the directed nor in the undirected sense, which is equivalent if every node has at most one incoming edge.

The announced class $\mathcal{R}(T_1, \widehat{t})$ consists of all instances I such that $I = I_{\mathcal{T}}$ for some $\widehat{t}$-proper T_1-labeled instance tree $\mathcal{T}$. It is easy to see that $\mathsf{chase}_{T_1}(\widehat{t}) \in \mathcal{R}(T_1, \widehat{t})$ as there is a $\widehat{t}$-proper T_1-labeled instance tree $\mathcal{T}$ such that $I_{\mathcal{T}} = \mathsf{chase}_{T_1}(\widehat{t})$. However, there are also instances $I \in \mathcal{R}(T_1, \widehat{t})$ that do not admit a homomorphism to $\mathsf{chase}(\widehat{t}, T_1)$. The following example illustrates their importance.

Example 3. *Consider $T_1, T_2, \Sigma_D, \Sigma_Q$ from Example 1 and I, D, c from Example 2. Let $\widehat{t} = \mathsf{tp}_{T_1}(\mathsf{chase}_{T_1}(D), c)$, that is, $\widehat{t} = \{A(x), \exists x\, A(x), \exists x\, B(x)\}$. Then $I \not\to \mathsf{chase}_{T_2}(\widehat{t})$. However, we find a $\widehat{t}$-proper T_1-labeled instance tree $\mathcal{T} = (V, E, B, \mu)$ such that $I \to I_{\mathcal{T}}$.*

We may construct $\mathcal{T}$ by starting with a single node v_0, $B(v_0) = \{R(c_1, c_0)\}$, and

$$\mu(c_0) = \mu(c_1) = \{B(x), \exists x\, A(x), \exists x\, B(x)\}.$$

Then repeatedly add a predecessor v_{i+1} of v_i, with $B(v_{i+1}) = \{R(c_{i+1}, c_i)\}$ and $\mu(c_{i+1}) = \mu(c_0)$, ad infinitum. The resulting tree $\mathcal{T}$ is $\widehat{t}$-proper and satisfies $I_{\mathcal{T}} = I$. Note that it does not have a root.

The next lemma is the core ingredient to the proof of Theorem 10. Informally, it states that when replacing $\mathsf{chase}_{T_1}(\widehat{t})$ with instances from $\mathcal{R}(T_1, \widehat{t})$, we may also replace homomorphism limits with homomorphisms.

Lemma 9. *Let I be a countable Σ-connected instance such that $\mathsf{adom}(I)$ contains only nulls. Then $I \to^{\mathsf{lim}}_{\Sigma} \mathsf{chase}_{T_1}(\widehat{t})$ iff there is an $\widehat{I} \in \mathcal{R}(T_1, \widehat{t})$ with $I \to \widehat{I}$.*

In the proof of Lemma 9, the laborious direction is 'only if', where one assumes that $I \to^{\mathsf{lim}}_{\Sigma} \mathsf{chase}_{T_1}(\widehat{t})$ and then uses finite subinstances $J_1 \subseteq J_2 \subseteq \cdots$ of I with $I = \bigcup_{i \geq 1} J_i$ and homomorphisms h_i from J_i to $\mathsf{chase}_{T_1}(\widehat{t})$, $i \geq 1$, to identify the desired instance $\widehat{I} \in \mathcal{R}(T_1, \widehat{t})$. This again involves several 'skipping homomorphisms' type of arguments.

Using Lemma 9, we give a decision procedure based on 2ATAs that establishes Theorem 10. The 2ATA accepts input trees encoding an instance $I \in \mathcal{R}(T_1, \widehat{t})$ that admits a Σ-homomorphism from $\mathsf{chase}_{T_2}(D_A)|^{\mathsf{con}}_{\Sigma}$.

7 Future Work

It would be interesting to determine the exact complexity of hom- and CQ-conservativity for frontier-one TGDs. We tend to think that these problems are 3EXPTIME-complete. Note that in the description logic $\mathcal{ELI}$, they are 2EXPTIME-complete (Jung et al. 2020).

It would also be interesting to study conservative extensions and triviality for other classes of TGDs that have been proposed in the literature. Of course, it would be of particular interest to identify decidable cases. Classes for which undecidability does not follow from the results in this paper include acyclic and sticky TGDs, which exist in several forms, see for instance (Calì, Gottlob, and Pieris 2010).

Acknowledgements

Carsten Lutz was supported by the DFG project LU 1417/3-1 QTEC and Jery Marcinkowski by the Polish National Science Centre (NCN) grant 2016/23/B/ST6/01438.

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

Artale, A.; Calvanese, D.; Kontchakov, R.; and Zakharyaschev, M. 2009. The DL-Lite family and relations. *J. Artif. Intell. Res.* 36:1–69.

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

Baget, J.; Mugnier, M.; Rudolph, S.; and Thomazo, M. 2011. Walking the complexity lines for generalized guarded existential rules. In *Proc. of IJCAI*, 712–717.

Benedikt, M.; Bourhis, P.; and Senellart, P. 2012. Monadic datalog containment. In *Proc. of ICALP*, volume 7392 of *LNCS*, 79–91. Springer.

Bienvenu, M., and Ortiz, M. 2015. Ontology-mediated query answering with data-tractable description logics. In *Reasoning Web*, 218–307.

Bienvenu, M.; ten Cate, B.; Lutz, C.; and Wolter, F. 2014. Ontology-based data access: A study through disjunctive Datalog, CSP, and MMSNP. *ACM Transactions on Database Systems* 39(4):33:1–33:44.

Botoeva, E.; Konev, B.; Lutz, C.; Ryzhikov, V.; Wolter, F.; and Zakharyaschev, M. 2016. Inseparability and conservative extensions of description logic ontologies: A survey. In *Proc. of Reasoning Web*, volume 9885 of *LNCS*, 27–89. Springer.

Botoeva, E.; Lutz, C.; Ryzhikov, V.; Wolter, F.; and Zakharyaschev, M. 2019. Query inseparability for $\mathcal{ALC}$ ontologies. *Artif. Intell.* 272:1–51.

Calì, A.; Gottlob, G.; Lukasiewicz, T.; Marnette, B.; and Pieris, A. 2010. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *Proc. of LICS*, 228–242. IEEE Computer Society.

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

Calì, A.; Gottlob, G.; and Lukasiewicz, T. 2012. A general datalog-based framework for tractable query answering over ontologies. *J. Web Semant.* 14:57–83.

Calì, A.; Gottlob, G.; and Pieris, A. 2010. Advanced processing for ontological queries. *Proc. VLDB Endow.* 3(1):554–565.

Calvanese, D.; Giacomo, G. D.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rodriguez-Muro, M.; and Rosati, R. 2009. Ontologies and databases: The DL-Lite approach. In *Reasoning Web*, volume 5689 of *LNCS*, 255–356.

Casanova, M. A.; Fagin, R.; and Papadimitriou, C. H. 1984. Inclusion dependencies and their interaction with functional dependencies. *J. Comput. Syst. Sci.* 28(1):29–59.

Conway, J. 1972. Unpredictable iterations. In *Proc. of 1972 Number Theory Conference*, 49–52.

Fagin, R.; Kolaitis, P. G.; Miller, R. J.; and Popa, L. 2005. Data exchange: semantics and query answering. *Theor. Comput. Sci.* 336(1):89–124.

Ghilardi, S.; Lutz, C.; and Wolter, F. 2006. Did I damage my ontology? A case for conservative extensions in description logics. In *Proc. of KR*, 187–197. AAAI Press.

Gogacz, T., and Marcinkowski, J. 2014. All-instances termination of chase is undecidable. In *Proc. of ICALP*, volume 8573 of *LNCS*, 293–304. Springer.

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. *Artif. Intell.* 213:42–59.

Gottlob, G.; Manna, M.; and Pieris, A. 2015. Polynomial rewritings for linear existential rules. In *Proc. of IJCAI*, 2992–2998. AAAI Press.

Gutiérrez-Basulto, V.; Jung, J. C.; and Sabellek, L. 2018. Reverse engineering queries in ontology-enriched systems: The case of expressive horn description logic ontologies. In *Proc. of IJCAI*, 1847–1853. ijcai.org.

Johnson, D. S., and Klug, A. C. 1984. Testing containment of conjunctive queries under functional and inclusion dependencies. *J. Comput. Syst. Sci.* 28(1):167–189.

Jung, J. C.; Lutz, C.; Martel, M.; Schneider, T.; and Wolter, F. 2017. Conservative extensions in guarded and two-variable fragments. In *Proc. of ICALP*, LIPIcs, 108:1–108:14.

Jung, J. C.; Lutz, C.; Martel, M.; and Schneider, T. 2018. Querying the unary negation fragment with regular path expressions. In *Proc. of ICDT*, volume 98 of *LIPIcs*, 15:1–15:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Jung, J. C.; Lutz, C.; Martel, M.; and Schneider, T. 2020. Conservative extensions in horn description logics with inverse roles. *J. Artif. Intell. Res.* 68:365–411.

Konev, B.; Kontchakov, R.; Ludwig, M.; Schneider, T.; Wolter, F.; and Zakharyaschev, M. 2011. Conjunctive query inseparability of OWL 2 QL TBoxes. In *Proc. of AAAI*. AAAI Press.

Lutz, C.; Walther, D.; and Wolter, F. 2007. Conservative extensions in expressive description logics. In *Proc. of IJCAI*, 453–458.

Ovid. 2008. *Metamorphoses, first edition 8 AD, Translated by A. D. Melville. Introduction and notes by Edward John Kenney.* Oxford University Press.