A New Decidable Class of Tuple Generating Dependencies: The Triangularly-Guarded Class

Vernon Asuncion and Yan Zhang
School of Computing, Engineering and Mathematics
Western Sydney University, Australia

Abstract
In this paper we introduce a new class of tuple-generating dependencies (TGDs) called triangularly-guarded TGDs, which are TGDs with certain restrictions on the atomic derivation track embedded in the underlying rule set. We show that conjunctive query answering under this new class of TGDs is decidable. We further show that this new class strictly contains some other decidable classes such as weak-acyclic, guarded, sticky and shy, which, to the best of our knowledge, provides a unified representation of all these aforementioned classes.

1 Introduction
In the classical database management systems (DBMS) setting, a query \( Q \) is evaluated against a database \( D \). However, it has come to the attention of the database community the necessity to also include ontological reasoning and description logics (DLs) along with standard database techniques [Calvanese et al., 2007]. As such, the ontological database management systems (ODBMS) has arised. In ODBMS, the classical database is enhanced with an ontology [Baader et al., 2016] in the form of logical assertions that generate new intensional knowledge. A powerful form of such logical assertions is the tuple-generating dependencies (TGDs), i.e., Horn rules extended by allowing existential quantifiers to appear in the rule heads [Cabinbo, 1998; Patel-Schneider and Horrocks, 2007; Cali et al., 2009].

Queries are evaluated against a database \( D \) and set of TGDs \( \Sigma \) (i.e., \( D \cup \Sigma \)) rather than just \( D \), as in the classical setting. Since for a given database \( D \), a set \( \Sigma \) of TGDs, and a conjunctive query \( Q \), the problem of determining if \( D \cup \Sigma \models Q \), called the conjunctive query answering (CQ-Ans) problem, is undecidable in general [Beeri and Vardi, 1981; Baget et al., 2011; Rosati, 2011; Cali et al., 2012; Cali et al., 2013], a major research effort has been put forth to identifying syntactic conditions on TGDs for which CQ-Ans is decidable. Through these efforts, we get the decidable syntactic classes: weakly-acyclic (WA) [Fagin et al., 2005], acyclic graph of rule dependencies (aGRD) [Baget et al., 2011], linear, multi-linear, guarded, weakly-guarded (W-GUARDED) [Rosati, 2006; Cali et al., 2013], sticky, sticky-join, weakly-sticky-join (WSJ) [Cali et al., 2012; Gogacz and Marcinkowski, 2017], shy (SHY) [Leone et al., 2012] and weakly-recursive (WR) [Civili and Rosati, 2012]. The weakly-recursive class is only defined for the so-called simple TGDs, which are TGDs where the variables are only allowed to occur once in each atom and each atom do not mention constants [Civili and Rosati, 2012].

Another research direction that sprangs up from those previously identified classes is the possibility of obtaining more expressive languages by a direct combination (i.e., union) of those classes, e.g., see [Krotzsch and Rudolph, 2011; Cali et al., 2012; Grau et al., 2013; Gottlob et al., 2013]

A major challenge then in this direction is that the union of two decidable classes is not necessarily decidable [Baget et al., 2011], e.g., it has been shown in [Gottlob et al., 2013] that the union of the classes linear and sticky is undecidable.

At a model theoretic level, the results in [Rosati, 2006; Barany et al., 2010] had respectively shown that the finite model property holds for the linear and guarded fragments of TGDs. It is folklore that a class of first-order (FO) theories \( \mathcal{C} \) is said to have the finite model (FM) property if \( \phi \in \mathcal{C} \) satisfiable iff \( \phi \) has a finite model. The recent work in [Gogacz and Marcinkowski, 2017] had further extended the result in [Rosati, 2006] for linear TGDs into the sticky-join TGDs. As will be revealed from this paper, our work further generalizes these previous results.

Despite these efforts, there are still some examples of simple TGDs that do not fall under the aforementioned classes.

Example 1. Let \( \Sigma_1 \) be a set of TGDs comprising of the following rules:

\[
\begin{align*}
\sigma_{11} & : \; t(X, Y) \land \exists Z \; t(Y, Z) \land u(Y, Z), \quad (1) \\
\sigma_{12} & : \; t(X, Y) \land u(Y, Z) \rightarrow t(Y, Z) \land u(X, Y). \quad (2)
\end{align*}
\]

Then it can be checked that \( \Sigma_1 \) does not fall into any of the classes previously mentioned above, and neither is it glut-guarded (G-GUARDED) [Krotzsch and Rudolph, 2011] nor tame (TAME) [Gottlob et al., 2013]. On the other hand, because none of the head atoms \( t(Y, Z) \) and \( u(X, Y) \) of \( \sigma_{12} \) mentions the two cyclically-affected body variables “X” and “Z” together (which is under some pattern that we will generalize in Section 3), then it can be shown that for any
where

### 2 Preliminaries

#### 2.1 Basic notions and notations

We assume three countably infinite pairwise disjoint sets \( \Gamma_\forall, \Gamma_\exists \) and \( \Gamma_N \) of variables, constants and labeled nulls, respectively. We further assume that \( \Gamma_\forall \) is partitioned into two disjoint sets \( \Gamma_\forall^0 \) and \( \Gamma_\forall^1 \) (i.e., \( \Gamma_\forall = \Gamma_\forall^0 \cup \Gamma_\forall^1 \)), where \( \Gamma_\forall^0 \) and \( \Gamma_\forall^1 \) denote the sets of universally \((\forall)\) and existentially \((\exists)\) quantified variables, respectively.

We also assume that the set of labeled nulls \( \Gamma_N \) contains elements of the form \( \{n_i \mid i \in \mathbb{N}\} \), where \( \mathbb{N} \) is the set of natural numbers. Intuitively, \( \Gamma_N \) is the set of “fresh” Skolem terms that are disjoint from the set of constants \( \Gamma_C \).

A relational schema \( \mathcal{R} \) (or just schema) is a set of relational symbols (or predicates), where each is associated with some number \( r \geq 0 \) called its arity. We denote by \( r/n \) as the relational symbol \( r \) with arity \( n \), and by \( |r| \) as the arity of \( r \), i.e., \( |r| = n \). We further denote by \( r[i] \) as the \( i \)-th argument (or attribute) of \( r \) where \( i \in \{0, \ldots, |r|\} \). We denote by \( \text{ARG}(r) \) as the set of arguments \( \{r[i] \mid i \in \{0, \ldots, |r|\}\} \) of \( r \). We extend this notion to the set of relational symbols \( \mathcal{R} \), i.e., \( \text{ARG}(\mathcal{R}) = \bigcup_{r \in \mathcal{R}} \text{ARG}(r) \).

A tuple \( t \) is any element from the set \( \Gamma_\forall \cup \Gamma_\exists \cup \Gamma_N \). Then an atom \( a \) is a construct of the form \( r(t_1, \ldots, t_n) \) such that:

1. \( r \in \mathcal{R} \);
2. \( n = |r| \); and
3. \( t_i \) (for \( i \in \{1, \ldots, n\} \)) is a term.

We denote tuples of atoms by \( \overline{a}, \) e.g., \( \overline{a} = a_1 \ldots a_i \). We denote atoms by \( \overline{\mathbb{N}} \).

We denote by \( \text{REL}(a), \text{TERMS}(a), \text{VAR}(a), \text{CONST}(a) \) and \( \text{NULLS}(a) \) as the relational symbol, the set of terms, variables, constants and labeled nulls mentioned in atom \( a \), respectively.

We extend this notion to sets or tuples of atoms \( S \) such that \( \text{TERMS}(S), \text{VAR}(S), \text{CONST}(S) \) and \( \text{NULLS}(S) \) denotes the sets \( \bigcup_{a \in S} \text{TERMS}(a), \bigcup_{a \in S} \text{VAR}(a), \bigcup_{a \in S} \text{CONST}(a) \) and \( \bigcup_{a \in S} \text{NULLS}(a) \), respectively. We say that a tuple of atoms \( \overline{a} = a_1 \ldots a_i \) is connected if either:

1. \( \overline{a} \) is an atom, or
2. \( \text{TERMS}(a_i) \cap \text{TERMS}(a_{i+1}) \neq \emptyset \) holds, for each \( i \in \{1, \ldots, l - 1\} \).

An instance \( I \) is any set (can be infinite) of atoms such that \( \text{VAR}(I) = \emptyset \), i.e., contains no variables. A database \( D \) is a finite set of ground atoms \( \text{VAR}(D) = \emptyset \) and \( \text{NULLS}(D) = \emptyset \).

Given an atom \( a = r(t_1, \ldots, t_n) \), we denote by \( \text{ARG}(a) \) as the set of arguments \( \{r[i] \mid i \in \{1, \ldots, n\}\} \). For a tuple of variables \( X \) and atom \( a = r(t_1, \ldots, t_n) \), we denote by \( \text{ARG}(a) \setminus X \) as the set of arguments \( \{r[i] \mid i \in \{1, \ldots, n\} \) and \( t_i = X_i \), i.e., the set of arguments in \( \text{ARG}(a) \) but restricted to those mentioned variables from \( X \). Symmetrically, using similar notions to the “\( \text{ARG} \)” concept just previously mentioned above, for a given atom \( a = r(t_1, \ldots, t_n) \) and set of arguments \( A \subseteq \text{ARG}(a) \), we denote by \( \text{VAR}(a) \setminus A \) as the set of variables \( \{X \mid t_i = X \) and \( r[i] \in A\} \), i.e., the set of all the variables mentioned in \( a \) but restricted to those appearing in argument positions from \( A \).

Given two sets of terms \( T_1 \) and \( T_2 \), an assignment \( \theta : T_1 \rightarrow T_2 \) is a function from \( T_1 \) onto \( T_2 \) such that \( t \in (T_1 \cap T_C) \) implies \( \theta(t) = t \), i.e., identity for the constants \( \Gamma_C \). Then for a given atom \( a = r(t_1, \ldots, t_n) \), a set of terms \( T \) and an assignment \( \theta : \text{VAR}(a) \rightarrow T \), a substitution of \( a \) under \( \theta \) (or just substitution for convenience), denoted \( a\theta \) (or sometimes \( \theta(a) \)), is the atom such that \( a\theta = r(\theta(t_1), \ldots, \theta(t_n)) \).

We naturally extend to conjunctions of atoms \( a_1 \wedge \ldots \wedge a_n \) so that \( \theta(a_1 \wedge \ldots \wedge a_n) = a_1 \theta \wedge \ldots \wedge a_n \theta \). Given two assignments \( \theta_1 : T_1 \rightarrow T_2 \) and \( \theta_2 : T_2 \rightarrow T_3 \), we denote by \( \theta_2 \circ \theta_1 \) as the composition of \( \theta_1 \) with \( \theta_2 \) such that \( \theta_2 \circ \theta_1 : T_1 \rightarrow T_3 \) and \( (\theta_2 \circ \theta_1)(t) = \theta_2(\theta_1(t)) \), for all \( t \in T_1 \). Then, finally, given an assignment \( \theta : T_1 \rightarrow T_2 \) and some set of terms \( T' \subseteq T_1 \), we denote by \( \theta|_{T'} \) as the restriction of the assignment \( \theta \) to the domain \( T' \subseteq T_1 \) such that \( \theta|_{T'}(t) = \theta(t) \in T_2 \), for each \( t \in T' \subseteq T_1 \).

### 2.2 TGDs, BCQ-Ans and Chase

A tuple generating dependency (TGD) rule \( \sigma \) of schema \( \mathcal{R} \) is a first-order (FO) formula of the form

\[
\forall X Y (\Phi(X, Y) \rightarrow \exists Z \Psi(Y, Z)),
\]

where:

- \( X = X_1 \ldots X_k \), \( Y = Y_1 \ldots Y_l \) and \( Z = Z_1 \ldots Z_m \) are pairwise disjoint tuple of variables, and where they are called the local, shared and existentinal variables, respectively, and thus, we assume that \( X \subseteq \Gamma_\forall \) and \( Z \subseteq \Gamma_\exists \);
- \( \Phi(X, Y) = b_1(V_1) \ldots b_n(V_n) \) is a conjunction of atoms such that \( V_i \subseteq XY \) and \( r_i \in R_i \), for \( i \in \{1, \ldots, n\} \);
- \( \Psi(Y, Z) = r_1(W_1) \ldots \ldots r_m(W_m) \) is a conjunction of atoms where \( W_i \subseteq YZ \) and \( r_i \in R_i \), for \( i \in \{1, \ldots, m\} \).

For a given TGD \( \sigma \) of the form \( \Phi(X, Y) \rightarrow \exists Z \Psi(Y, Z) \), we denote by \( \text{BD}(\sigma) \) as the set of atoms \( \{b_1(V_1), \ldots, b_n(V_n)\} \), which we also refer to as the **body** of \( \sigma \). Similarly, by \( \text{HD}(\sigma) \) we denote the set of atoms \( \{r_1(W_1), \ldots, r_m(W_m)\} \), which we also refer to as the **head** of \( \sigma \). For convenience, when it is clear from the context, we simply drop the quantifiers in \( \Phi(X, Y) \rightarrow \exists Z \Psi(Y, Z) \) such that a TGD rule \( \sigma \) of the form \( \Phi(X, Y) \rightarrow \exists Z \Psi(Y, Z) \) can simply be referred to as: \( \Phi(X, Y) \rightarrow \Psi(Y, Z) \). Then, for a given set of TGDs \( \Sigma \), we denote by \( \text{ATOMS}(\Sigma) \) as the set of all atoms occurring in \( \Sigma \) such that \( \text{ATOMS}(\Sigma) = \bigcup_{\sigma \in \Sigma} (\text{BD}(\sigma) \cup \text{HD}(\sigma)) \), and by \( \text{REL}(\Sigma) \) as the set of all relational symbols mentioned in \( \Sigma \). Then, lastly, for convenience later on, for a given rule \( \sigma \) of the form \( \Phi(X, Y) \rightarrow \exists Z \Psi(Y, Z) \) and atom \( a \in \text{HD}(\sigma) \), we denote by \( \forall\text{VAR}(a) \) and \( \exists\text{VAR}(a) \) as
the set of variables $\text{VAR}(a) \cap Y$ and $\text{VAR}(a) \cap Z$, respectively, i.e., the set of all the universally ($\forall$) and existentially ($\exists$) quantified variables of $a$, respectively. We extend this notion to the TGD rule $\sigma$ of the form $\exists$, so that we set $\text{VAR}(\sigma) = \text{XY}Z$, $\forall\text{-VAR}(\sigma) = \text{XY}$ and $\exists\text{-VAR}(\sigma) = \text{Z}$.

A boolean conjunctive query (BCQ) $Q$ is a FO formula $\exists X \varphi(X) \rightarrow q$ such that $\varphi(X) = r_1(Y_1) \land \ldots \land r_n(Y_n)$, where $r_i \in \mathcal{R}$ and $Y_i \subseteq X$, for each $i \in \{1, \ldots, n\}$, and where we set $\text{BD}(Q) = \{r_1(Y_1), \ldots, r_n(Y_n)\}$. Given a database $D$ and a set of TGDs $\Sigma$, we say that $D \cup \Sigma$ entails $Q$, denoted $D \cup \Sigma \models Q$, iff $D \cup \Sigma \models \exists X \varphi(X)$. The central problem tackled in this work is the boolean conjunctive query answering (BCQ-Ans): given a database $D$, a set of TGDs $\Sigma$ and BCQ $Q$, does $D \cup \Sigma \models Q$? It is well known that BCQ-Ans is undecidable in general [Beeri and Vardi, 1981].

The chase procedure (or just chase) [Maier et al., 1979; Johnson and Klug, 1984; Fagin et al., 2005; Deutsch et al., 2008; Zhang et al., 2015] is a main algorithmic tool proposed for checking implication dependencies [Maier et al., 1979]. For an instance $I$, assignment $\eta$ and TGD $\sigma = \Phi(X, Y, Z)$, we have that $I \models \sigma, I'$ defines a simple chase step as follows: $I' = I \cup \{\eta(\Phi(X, Y))\} \cup I_f$ such that: (1) $\eta: \text{XY} \rightarrow \Gamma_c \cup \Gamma_n$ and $\eta(\Phi(X, Y)) \subseteq I_f$; and (2) $\eta': \text{XY} \rightarrow \Gamma_c \cup \Gamma_n$ and $\eta'_{|\text{XY}} = \eta$. As in the literatures, we further assume here that each labeled nulls used to eliminate the $\exists$-quantified variables in $Z$ follows lexicographically all the previous ones, i.e., follows the order $n_1, n_1+1, n_1+2, \ldots$. A chase sequence of a database $D$ wrt. to a set of TGDs $\Sigma$ is a sequence of chase steps $I_i = I_{i-1} \cup \sigma \models_a \text{nulls}_a \models \cdots \models I_{i+1}$, where $i \geq 0$, $I_0 = D$ and $\sigma \in \Sigma$. An infinite chase sequence $I_i = I_{i+1}$ is fair if $\eta(\Phi(X, Y)) \subseteq I_f$ for some $\eta: \text{XY} \rightarrow \Gamma_c \cup \Gamma_n$ and $\sigma = \Phi(X, Y)$.

Given a database $D$, TGDs $\Sigma$ and BCQ $Q$, $D \cup \Sigma \models Q$ iff chase($D, \Sigma$) = $\exists$.

2.3 Cyclically-affected arguments

As observed in [Leone et al., 2012], the notion of affected arguments in [Calli et al., 2013] can sometimes consider arguments that may not actually admit a “firing” mapping $\forall$-variables into nulls. For this reason, it was introduced in [Leone et al., 2012] the notion of a “null-set.” Given a set of TGDs $\Sigma$, let $a \in \text{ATOMS}(\Sigma)$, $a \in \text{ARG}(a)$ and $X = \text{VAR}(a)$, then the null-set of $a$ under $\Sigma$, denoted as $\text{nullset}(\Sigma, a, X)$ (or just $\text{nullset}(a, X)$ if clear from the context), is defined inductively as follows: If $a \in \text{HD}(\sigma)$, for some $\sigma \in \Sigma$, then: (1) $\text{nullset}(a, X) = \{n_1^X, n_2^X\}$ if $\text{VAR}(a) = X$; (2) $\text{nullset}(a, X)$ is the intersection of all null-sets nullset($b$, $b$) such that $b \in \text{BD}(\sigma)$, $b \in \text{ARG}(b)$ and $\text{VAR}(b) = X$. Otherwise, if $a \in \text{BD}(\sigma)$, for some $\sigma \in \Sigma$, then $\text{nullset}(a, a)$ is the union of all $\text{nullset}(a, a')$ such that $\text{REL}(a') = \text{REL}(a)$ and $a' \in \text{HD}(\sigma')$, where $\sigma' \in \Sigma$.

Borrowing similar notions from [Krotzsch and Rudolph, 2011] used in the identification of the so-called glut variables, the existential dependency graph $\mathcal{G}_2(\Sigma)$ is a graph $(N, E)$, whose nodes $N$ is the union of all $\text{nullset}(a, a)$, where $a \in \text{ATOMS}(\Sigma)$ and $a \in \text{ARG}(a)$, and edges:

$E = \{ (n_2^a, n_2^b) \mid \exists \sigma \in \Sigma \text{ of form } \exists, \forall Y \in Y, n_2^a \in \bigcap \text{nullset}(Y, \sigma, \Sigma) \text{ and } \text{nullset}(a, a) = \{n_2^a\},$ for some $a \in \text{HD}(\sigma)$ and $Z \in Z \}$

where $\bigcap \text{nullset}(Y, \sigma, \Sigma)$ denotes the intersection of all $\text{nullset}(b, b, \Sigma)$ such that $b \in \text{BD}(\sigma)$, $b \in \text{ARG}(b)$ and $Y = \text{VAR}(b)$, $b$. We note that our definition of a dependency graph here generalizes the existential dependency graph in [Krotzsch and Rudolph, 2011] by combining the notion of null-sets in [Leone et al., 2012]. Then with the graph $\mathcal{G}_2(\Sigma) = (N, E)$ as defined above, we denote by $\text{CYC-NULL}(\Sigma)$ as the smallest subset of $N$ such that $n_2^a \in \text{CYC-NULL}(\Sigma)$ iff either: (1) $n_2^a$ is in a cycle in $\mathcal{G}_2(\Sigma)$, (2) $n_2^a$ is reachable from some other node $n_2^b$, $b \in \text{CYC-NULL}(\Sigma)$, where $n_2^a$ is in a cycle in $\mathcal{G}_2(\Sigma)$.

3 Triangularly-Guarded (TG) TGDs

This section now introduces the triangularly-guarded class of TGDs, which is then used. We begin with an instance of a BCQ-Ans problem that corresponds to a need of an infinite number of labeled nulls in the underlying chase derivation.

Example 2 (Unbounded nulls). Let $\Sigma_2 = \{\sigma_{11}, \sigma_{12}\}$ be the set of TGDs obtained from $\Sigma_1 = \{\sigma_{11}, \sigma_{12}\}$ of Example[Calli et al., 2013] by just changing the rule $\sigma_{12}$ into the rule $\sigma'_{12}$ such that:

$\sigma'_{12} \models t(X, Y) \land \forall u(Y, Z) \rightarrow t(X, Z) \land \forall u(X, Y).$ (4)

Then we have that $\sigma'_{12}$ of $\Sigma_2$ above is obtained from $\sigma_{12}$ of $\Sigma_1$ by changing the variable “$Y$” in the head atom “$t(X, Y)$” of $\sigma_{12}$ into “$X$”, i.e., to obtain “$t(X, Z)$”. Intuitively, this allows the two variables “$X$” and “$Y$” to act as place holders that combines labeled nulls together in the head atom “$t(X, Z)$” of $\sigma_{12}$. Now let $D_2 = \{t(c_1, c_2), u(c_1, c_2)\}$ be a database, where $c_1, c_2 \in \text{GC}_c$ and $c_1 \neq c_2$, and $Q_2$ the BCQ $\exists X t(X, Y) \rightarrow q$. Then we have that $D_2 \cup \Sigma_2 \cup \{t(X, X)\}_{X} \models D_2 \cup \Sigma_2 \cup \{\forall X \exists t(X, X)\}$ can only be satisfied by the infinite model $M$ of the form $M = D_2 \cup \bigcup_{i < j} \{t(c_i, c_j)\}$, where we assume $i \neq j$ implies $c_i \neq c_j$. Therefore, since $D_2 \cup \Sigma_2 \cup \{\forall X \neg t(X, X)\}$ is satisfiable, (albeit infinitely), it follows that $D_2 \cup \Sigma_2 \not\models Q_2$.

In database $D_2$ and set TGDs $\Sigma_2 = \{\sigma_{11}, \sigma'_{12}\}$ of Example[Calli et al., 2013] we get from $\sigma_{11}$ the sequence of atoms $t(c_2, n_1), t(n_1, n_2), t(n_2, n_3), \ldots, t(n_{k-1}, n_k) \in \text{chase}(D_2, \Sigma_2)$, where $n_i \in \text{IN}_n$, for each $i \in \{1, \ldots, k\}$. Moreover, by the repeated applications of $\sigma'_{12}$, we further get that $t(n, n_k) \in \text{chase}(D_2, \Sigma_2)$, for each $i \in \{1, \ldots, k-1\}$, i.e., $n_i$ and $n_k$, for each $i \in \{1, \ldots, k-1\}$, will be “pulled” together in some relation
of \( t \) in \( \text{chase}(D_2, \Sigma_2) \). As such, for the given BCQ \( Q_2 = \exists X (X, X) \rightarrow q \) also from Example 2, since \( D_2 \cup \Sigma_2 \models Q_2 \) iff \( D_2 \cup \Sigma_2 \cup \{ X \rightarrow t(X, X) \} \) is not satisfiable, then the fact that we have to satisfy the literal \( \neg t(X, X) \) for all \( \forall X \), and because \( t(n_1, n_k) \in \text{chase}(D_2, \Sigma_2) \), for \( i \in \{1, \ldots, k-1\} \), implies that each of the \( n_i \) must be of different values from \( n_k \), and thus cannot be represented by a finite number of distinct labeled nulls.

### 3.1 Triangular-components of TGD extensions

In contrast to \( \Sigma_2 \) from Example 2, what we aim to achieve now is to identify syntactic conditions on TGDs so that such a “distinguishable” of labeled nulls is limited in the chase derivation. As a consequence, we end up with some nulls that need not be distinguishable from another, and as such, we can actually re-use these nulls without introducing new ones in the chase derivation. This leads BCQ-Ans to be decidable.

**Definition 1 (TGD extension).** Given a set of TGDs \( \Sigma \), we denote by \( \Sigma^+ \) as the extension of \( \Sigma \), and is inductively defined as follows:

\[
\Sigma^0 = \{ \langle BD(\sigma), HD(\sigma) \rangle \mid \sigma \in \Sigma \};
\]

\[
\Sigma^{i+1} = \Sigma^i \cup \{ \langle B_1 H_1 \cup B_2 H_2 \rangle, (B_1, H_1) \in \Sigma^i, (B_2, H_2) \in \Sigma^i \}
\]

where \( \eta_1 : \text{VAR}(B_1 \cup H_1) \rightarrow \text{TERMS}(B_1 \cup H_1) \) and \( \eta_2 : \text{VAR}(B_2 \cup H_2) \rightarrow \text{TERMS}(B_1 \eta_1 \cup B_2 \theta \cup H_2 \theta) \) such that:

1. \( \theta \) is a renaming (bijective) substitution such that \( \text{VAR}(B_1 \cup H_1) \cap \text{VAR}(B_2 \theta \cup H_2 \theta) = \emptyset \);
2. \( \exists H_1' \subseteq H_1 \) and \( \exists B_2' \subseteq B_2 \) such that \( H_1' \eta_1 = B_2' \eta_2 \) corresponds to the MGU of \( H_1' \) and \( B_2' \), and
3. \( B^* = B_2(\eta_2 \theta \setminus B_2' \eta_2) \).

Then we set \( \Sigma^+ = \Sigma^\infty \) as the fixpoint of \( \Sigma^i \). We note that even though \( \Sigma^+ \) can be infinite in general, it follows from Theorem 5 that it is enough to consider a finite number of iterations \( \Sigma^i \) to determine “recursive triangular-components” (as will be defined exactly in Definition 2).

The TGD extension \( \Sigma^+ \) of \( \Sigma \) contains as elements pairs of sets of atoms of the form \( \{B, H\} \). Loosely speaking, the set \( B \) represents the body of some TGD in \( \Sigma \) while \( H \) as the head atoms that can be linked (transitive) through the repeated applications of the steps in (6) (which is done until a fixpoint is reached). The base case \( \Sigma^0 \) in (5) first considers the pairs \( \{B, H\} \), where \( B = \text{BD}(\sigma) \) and \( H = \text{HD}(\sigma) \), for each \( \sigma \in \Sigma \). Inductively, assuming we have already computed \( \Sigma^i \), we have that \( \Sigma^{i+1} \) is obtained by adding the previous step \( \Sigma^{i-1} \) as well as adding the set as defined through (6) (7).

More specifically, using similar ideas to the TGD expansion in [Call et al., 2012] that was used in identifying the sticky-join class of TGDs and tame reachability in [Gottlob et al., 2013] used for the tame class, the set (6) - (7) considers the other head types that can be (transitively) reached from some originating TGD. Indeed, as described in (6) - (7), for \( \langle B_1, H_1 \rangle \in \Sigma^1 \) and \( \langle B_2, H_2 \rangle \in \Sigma^1 \), we add the pair \( \{B_1 \eta_1 \cup B_2^*, H_2(\eta_2 \theta)\} \) into \( \Sigma^{i+1} \). Intuitively, with the assignment “\( \eta_2 \theta \sigma \) as described in (6) - (7), the aforementioned pair \( \{B_1 \eta_1 \cup B_2^*, H_2(\eta_2 \theta)\} \) encodes the possibility that “\( H_2(\eta_2 \theta) \)” can be derived transitive from bodies \( B_1 \eta_1 \) and \( B^* = B_2(\eta_2 \theta \setminus B_2' \eta_2) \). We note that the renaming function “\( \eta^* \)” is only used for pair \( \{B_2, H_2\} \) in (6) - (7) and (no renaming for pair \( \{B_1, H_1\} \)) so that we can track some of the originating variables from \( B_1 \) all the way through the head “\( H_2(\eta_2 \theta) \)”, and which can be retained through iterative applications of the criterion given in (6) - (7). Importantly, we note that the connection between \( B_1 \) and \( H_2 \) is inferred with \( H_1' \eta_1 = B_2' \eta_2 \) (where \( H_1' \subseteq H_1 \) and \( B_2' \subseteq B_2 \)) corresponding to the most general unifier (MGU) of \( H_1' \) and \( B_2' \) (please see Condition (2) of set (6) - (7)).

**Example 3.** Let \( \Sigma_3 \) be the following set of TGD rules:

\[
\sigma_{31} : t(X, Y) \rightarrow \exists Z t(Y, Z),
\]

\[
\sigma_{32} : t(X, Y) \rightarrow s(X) \land s(Y),
\]

\[
\sigma_{33} : t(X_1, V) \land s(V) \land t(W, Z_1) \rightarrow u(X_1, V, W, Z_1),
\]

\[
\sigma_{34} : u(X_2, Y, Z_2) \rightarrow v(X_2, Z_2),
\]

\[
\sigma_{35} : v(X_3, Z_3) \rightarrow t(X_3, Z_3).
\]

Then from the rules \( \sigma_{33}, \sigma_{34}, \sigma_{35} \), we get the three pairs

\[
\rho_{31} = \{ \{ t(X_1, V), s(V), t(W, Z_1) \}, \{ u(X_1, V, W, Z_1) \} \},
\]

\[
\rho_{32} = \{ \{ u(X_2, Y, Z_2) \}, \{ v(X_2, Z_2) \} \},
\]

\[
\rho_{33} = \{ \{ v(X_3, Z_3) \}, \{ t(X_3, Z_3) \} \} \text{ in } \Sigma_3^{\infty}, \text{ respectively.}
\]

Then finally, through the unification of the head atom “\( u(X_1, V, W, Z_1) \)” of rule \( \sigma_{33} \) with the body atom “\( u(X_2, Y, Z_2) \)” of \( \sigma_{34} \), we get the pair \( p_{31} = \{ \{ t(X_1, V), s(V), t(W, Z_1) \}, \{ v(X_2, Z_2) \} \} \) \subseteq \Sigma_3^{\infty}. \text{ Then finally, through the unification of the head atom “} v(X_2, Z_2) \text{” of } \sigma_{34} \text{ and the body atom “} v(X_3, Z_3) \text{” of } \sigma_{35} \text{, we further get the pair } p_{32} = \{ \{ t(X_1, V), s(V), t(W, Z_1) \}, \{ v(X_2, Z_2) \}, \{ v(X_3, Z_3) \} \} \subseteq \Sigma_3^{\infty}. \text{ As will be seen in Definition 2 the last pair } p_{32} \text{ in Example 3 corresponds to what we will call a “recursive triangular-component” that will be defined precisely in Definition 2.}

### 3.2 Triangularly-guarded TGDs

In this section, we now introduce the key notion of triangularly-guarded TGDs, which are the triangular-components. It will first be necessary to introduce the following notions of cyclically-affected only and link variables of body atoms, as well as variable markups that borrows some concepts from [Call et al., 2012].

We first introduce the notion of cyclically-affected only variables in the body (i.e., set \( B \)) of some pair \( \langle B, H \rangle \) in \( \Sigma^+ \) where \( \Sigma \) is a set of TGDs. So towards this purpose, for a given pair \( \langle B, H \rangle \) in \( \Sigma^+ \), we define \( \text{VAR}^*(\Sigma, B) \) (i.e., \( \text{VAR}^* \) is read var-hat) as the set of variables: \( \{ X \mid X \in \text{VAR}(B) \) and \( \bigcap \text{NULLSET}(X, \sigma, \Sigma)[B] \cap \text{CYC-NULL}(\Sigma) \neq 0 \} \), where \( \bigcap \text{NULLSET}(X, \sigma, \Sigma)[B] \) denotes the intersection of the unions \( \bigcup_{b \in \text{ARG}(b')} b' \in \text{EHD}(b'), \sigma \in \Sigma \) \text{NULLSET}(b, b', \Sigma) \) for each pair \( (b, b') \text{ such that } b \in \text{ARG}(b) \) and \( b' \in B \). For convenience and when clear from the context, we simply refer to \( \text{VAR}^*(\Sigma, B) \) as \( \text{VAR}(B) \). Intuitively, variables in
where:

Then a set of TGDs and \( \Sigma \) cyclically-affected only variables of \( H \) set of variables in the intersections \( (\var{B_1} \cap \var{B_2}) \) (or just \( \text{LINK}(B, b_1, b_2) \) when clear from the context) as the set of variables in the intersections \( (\var{B_1} \cap \var{B_2}) \) Intuitively, \( \text{LINK}(B, b_1, b_2) \) denotes the cyclically-affected only variables of \( B \) that can actually “join” (link) two common nulls between the body atoms \( b_1 \) and \( b_2 \) that can be obtained through some firing substitution.

Lastly, we now introduce the notion of variable markup. Let \( a, c \) and \( a' \) be three atoms such that \( \text{REL}(a) = \text{REL}(a') \). Then similarly to [Cali et al., 2012], we define the “markup procedure” as follows. For the base case, we let \( a^0 \) (resp. \( c^0 \)) denote the atom obtained from \( a \) (resp. \( c \)) by marking each variable \( X \in \var{a} \) (resp. \( X \in \var{c} \)) such that \( X \notin \var{c} \) (resp. \( X \notin \var{a} \)). Inductively, we define \( a^{i+1} \) (resp. \( c^{i+1} \)) to be the atom obtained from \( a^i \) (resp. \( c^i \)) as follows: for each variable \( X \in \var{c} \) (resp. \( X \in \var{a} \)), if each variables in positions \( \text{ARG}(c)/X \) (resp. \( \text{ARG}(a')/X \)) occurs as marked in \( c \) (resp. \( a' \)), then each occurrence of \( X \) is marked in \( c \) (resp. \( c' \)) to obtain the new atom \( c^{i+1} \) (resp. \( a'^{i+1} \)). Then naturally, we denote by \( a^\infty \) (resp. \( c^\infty \)) as the fixpoint of the markup applications. Finally, we denote by \( M \var{a, c, a'} \) as the set of all the marked variables mentioned only in \( a^\infty \) under atoms \( a \) and \( a' \) as obtained through the method above.

Loosely speaking, in the aforementioned variable markup above, we can think of \( a \) as corresponding to some “body atom” while \( c \) and \( a' \) as “head atoms” that are reachable through the TGD extension \( \Sigma^+ \) (see Definition 1) as will respectively occur in some derivation track. Intuitively, the marked variables represent element positions that may fail the “sticky-join” property, i.e., disappear in the derivation track. Intuitively, the sticky-join property insures decidability because only a finite number of elements can circulate among the derivation tracks. As will be revealed in following Definition 2 we further note that we only consider marked variables in terms of the triple \( \langle a, c, a' \rangle \) because we only consider them for “recursive triangular-components.”

**Definition 2 (Recursive triangular-components).** Let \( \Sigma \) be a set of TGDs and \( \Sigma^+ \) its extension as defined in Definition 1 Then a recursive triangular-component (RTC) \( T \) is a tuple \[(B, H), \{a, b, c\}, \langle X, Z \rangle, a', \]
where: 1) \( (B, H) \in \Sigma^+ \); 2) \( \{a, b\} \subseteq B, a \neq b \) and \( c \in H \); 3) \( a' \) is an atom and there exists an assignment \( \theta : \var{a} \rightarrow \var{a'} \) such that \( a\theta = a' \) and either one of the following holds:
(a) \( c = a' \), or
(b) there exists \( (B', H') \in \Sigma^+ \) and function \( \eta : \var{B', H'} \rightarrow \Gamma C \cup \Gamma Y \), where \( \theta' \) is just a renaming substi-

ution such that \( \var{B \theta' \cup H' \theta'} \cap \var{B \cup H} = \emptyset \), and where \( c \in B' (\eta \circ \theta') \) and \( a' \in H' (\eta \circ \theta') \):

4) \( X \) and \( Z \) are two distinct variables where \( \{X, Z\} \subseteq \var{B} \), and \( X \in \var{a} \), \( Z \in \var{b} \), \( \{X, Z\} \subseteq \var{c} \) and \( X \in \var{a'} \); and lastly, 5) there exists a tuple of distinct atoms \( d_1, \ldots, d_m \subseteq B \) such that:
(a) \( a = d_1 \) and \( d_m \), and for each \( i \in \{1, \ldots, m - 1\} \), there exists \( Y_i \in \text{LINK}(B, d_i, d_{i+1}) \);
(b) for some \( i \in \{1, \ldots, m - 1\} \), there exists \( Y' \in \text{LINK}(B, d_i, d_{i+1}) \) such that \( Y' \in \var{a'} \)

Loosely speaking, a recursive triangular-component (RTC) \( T \) of the form (8) (see Definition 2 and Figure 1), can possibly enforce an infinite cycle of labeled nulls being “pulled” together into a relation in the chase derivation. We explain this by using again the TGDs \( \Sigma_2 = \{\sigma_{11}, \sigma_{12}\} \) and database \( D_2 \) of Example 2 Here, we let assume that \( B = BD(\sigma_{12}) \) and \( H = HD(\sigma_{12}) \) such that \( (B, H) \) is the pair mentioned (8). Then with the body atoms \( t(X, Y), u(Y, Z) \in BD(\sigma_{12}) \) and head atom \( t(X, Z) \in HD(\sigma_{12}) \) also standing for the atoms \( a, b \) and \( c \) in (3), respectively, then we can form the RTC: \[(t(B, H), (t(X, Y), u(Y, Z), t(X, Z)), \langle X, Z \rangle) \] (9)

We note here from Condition 3) of Definition 2 that the atom \( c' \) in (3) is also the head atom “t(X, Z),” i.e., the choice \( a = c' \) of Condition 3) holds in this case. For simplicity, we note that out example RTC in (9) retains the names of the variables “X” and “Y” mentioned in (8). Loosely speaking, for two atoms \( \{n_1, n_2\}, \{n_1, n_2\} \in \text{chase}(D_2, \Sigma_2) \), we have that rule \( \sigma_{12} \) and its head atom “t(X, Z)” would combine the two nulls “n1” and “n2” into a relation “\{n1, n2\}” in \( \text{chase}(D_2, \Sigma_2) \). Since the variable “X” is retained in each RTC cycle via Condition 4) (see Figure 1), this makes possible that nulls held by “X” in each cycle (in some substitution) to be pulled together into some other nulls held by “Z” as derived through the head atom “t(X, Z)”.

We further note that the connecting variable “Y” between the two body atoms “t(X, Y)” and “u(Y, Z)” corresponds to the variables \( Y_i \in \text{LINK}(B, d_i, d_{i+1}) \) of point (a) of Condition 5), and for some \( i \), some \( Y' \in \text{LINK}(B, d_i, d_{i+1}) \) also appears as marked (i.e., \( Y' \in M \var{a, c, a'} \)) in point (b) of Condition 5) with respect to the atom \( a' \). Intuitively, we require in (b) of Condition 5) that some of these variables \( Y' \) occur as marked (w.r.t. \( a' \)) so that labeled nulls of some link variables have a chance to disappear in the RTC cycle for otherwise, they can only link and combine a bounded number of labeled nulls due to the sticky-join property [Cali et al., 2012].

**Example 4.** Consider again the pair \( p_{21} = \{t(X_1, V), s(V), t(V, Z_1)\} \in \Sigma^3 \) from Example 3 Then with the pair \( p_{21} \) standing for \( (B, H) \) in (3) the atoms “t(X_1, V),” “t(V, Z_1),” “t(X_1, Z_1)” and “t(X_1, Z_1)” for the atoms \( a, b, c \) and \( c' \) in (5), respectively, and variables \( \{X_1, Z_1\} \) for the variables \( \langle Z, X \rangle \) in (5), then we can get a corresponding RTC \( T_1 = \{p_{21}, \langle t(X_1, V), t(V, Z_1), t(X_1, Z_1), t(X_1, Z_1)\rangle \} \) as illustrated in Figure 2.
Before we present the following Theorem, it is necessary to firstly introduce the notion level in a chase that we define inductively as follows \cite{cali2012}. (1) for an atom \(a \in D\), we set \(\text{LEVEL}(a) = 0\); then inductively, (2) for an atom \(a \in \text{chase}(D, \Sigma)\) obtained via some chase step \(I_k \xrightarrow{\sigma, \eta} I_{k+1}\), we set \(\text{LEVEL}(a) = \max\{\text{LEVEL}(b) \mid b \in \text{BD}(\eta)\} + 1\). Then finally, for some given \(k \in \mathbb{N}\), we set \(\text{chase}^k(D, \Sigma) = \{a \mid a \in \text{chase}(D, \Sigma) \text{ and LEVEL}(a) \leq k\}\). Intuitively, \(\text{chase}^k(D, \Sigma)\) is the instance containing atoms that can be derived in a fewer or equal to \(k\) chase steps.

**Theorem 2 (Bounded nulls).** Let \(D\) be a database and \(\Sigma \in \text{TG}\). Then for each tuple of atoms \(\vec{a}\), \(\exists N \in \mathbb{N}\) such that \(\forall k \in \mathbb{N}\), we have that \(n_j \in \left(\text{NULLS} (\text{chase}^{N+k}(D, \Sigma)) \setminus \text{NULLS} (\text{chase}^N(D, \Sigma))\right)\) implies \(\exists n_i \in \text{NULLS} (\text{chase}^N(D, \Sigma))\) where \(n_i\) and \(n_j\) are \(\vec{a}\)-interchangeable under \(\text{chase}(D, \Sigma)\).

**Proof (Sketch).** A contradiction can be derived by assuming that \(\exists \vec{a}, \forall N \in \mathbb{N}, \exists k \in \mathbb{N}, \exists n_j \in \Gamma_N, \forall n_i \in \Gamma_N\), where: \(n_j \in \left(\text{NULLS} (\text{chase}^{N+k}(D, \Sigma)) \setminus \text{NULLS} (\text{chase}^N(D, \Sigma))\right)\) implies the existence of an infinite distinguishing relation among all those nulls \(n_i\) and \(n_j\). Therefore, it follows that there must exists some RTC \(T\) of the form \(\Sigma_3\) where there are no body atom \(d\) that guards variables “\(X\)” and “\(Z\)”, i.e., \(\{X, Z\} \subseteq \text{VAR}(d)\) (see Definition 3).

**Theorem 3 (Finite model property).** For database \(D\), TGDs \(\Sigma \in \text{TG} \land \text{BCQ} Q, D \cup \Sigma \cup \{\neg Q\}\) have the FM property.

**Proof (Sketch).** By Lemma 2 each of the null \(n_j \in \left(\text{NULLS} (\text{chase}^{N+k}(D, \Sigma)) \setminus \text{NULLS} (\text{chase}^N(D, \Sigma))\right)\) is always \(\vec{a}\)-interchangeable with some null \(n_i \in \text{NULLS}(\text{chase}^N(D, \Sigma))\). Then it follows that \(\text{chase}(D, \Sigma)\) can be represented by a finite number of nulls from which the finite model property follows.

**Theorem 4 (Comparison with other syntactic classes).** For each class \(C \in \{\text{WA, W-GUARDED, WSJ, G-GUARDED, SHY, TAME, WR}\}\), we have that \(C \subseteq \text{TG}\).

**Proof (Sketch).** A contradiction is derived by assuming that \(C \in \{\text{WA, W-GUARDED, WSJ, G-GUARDED, SHY, TAME, WR}\}\) but where \(C \notin \text{TG}\), since we have by Definition 3 that \(C \notin \text{TG}\) implies that there exists some RTC where the variables \(X\) and \(Z\) are not guarded by some atom \(d \in B\).

**Theorem 5 (Computational complexities).** (1) Determining if \(\Sigma \in \text{TG}\) is in 2-EXPTIME (upper-bound) but is PSPACE-hard (lower-bound); (2) The BCQ-Ans combined complexity problem under the class TG is in 4-EXPTIME (upper-bound) but is 3-EXPTIME-hard (lower-bound).

In this paper, we have introduced a new class of TGDs called triangularly-guarded TGDs (TG), for which BCQ-Ans is decidable as well as having the FM property (Theorems 2 and 3). We further showed that TG strictly contains the
current main syntactic classes: WA, W-GUARDED, WSI, G-GUARDED, SHY, TAME and WR (Theorem [4], which, to the best of our knowledge, provides a unified representation of those aforementioned TGD classes.

References

[Abiteboul et al., 1995] Serge Abiteboul, Richard Hull, and Victor Vianu. Foundations of Databases. Addison-Wesley, 1995.

[Baader et al., 2016] Franz Baader, Meghyn Bienvenu, Carsten Lutz, and Frank Wolter. Query and predicate emptiness in ontology-based data access. J. Artif. Intell. Res., 56:1–59, 2016.

[Baget et al., 2011] Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat. On rules with existential variables: Walking the decidability line. Artif. Intell., 175(9-10):1620–1654, 2011.

[Bárány et al., 2010] Vince Bárány, Georg Gottlob, and Martin Otto. Querying the guarded fragment. In Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom, pages 1–10, 2010.

[Beeri and Vardi, 1981] Catriel Beeri and Moshe Y. Vardi. The implication problem for data dependencies. In Automata, Languages and Programming, 8th Colloquium, Acre (Akko), Israel, July 13-17, 1981, Proceedings, pages 73–85, 1981.

[Cabibbo, 1998] Luca Cabibbo. The expressive power of stratified logic programs with value invention. Inf. Comput., 147(1):22–56, 1998.

[Cali et al., 2009] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. In Proceedings of the Twenty-Eighth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009, June 19 - July 1, 2009, Providence, Rhode Island, USA, pages 77–86, 2009.

[Cali et al., 2012] Andrea Calì, Georg Gottlob, and Andreas Pieris. Towards more expressive ontology languages: The query answering problem. Artif. Intell., 193:87–128, 2012.

[Cali et al., 2013] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. J. Artif. Intell. Res., 48:115–174, 2013.

[Calvanese et al., 2007] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. Autom. Reasoning, 39(3):385–429, 2007.

[Civili and Rosati, 2012] Cristina Civili and Riccardo Rosati. A broad class of first-order rewritable tuple-generating dependencies. In Proceedings of the 2nd International Conference on Datalog in Academia and Industry (Datalog-2012), pages 68–80, 2012.

[Deutsch et al., 2008] Alin Deutsch, Alan Nash, and Jeffrey B. Remmel. The chase revisited. In Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008, June 9-11, 2008, Vancouver, BC, Canada, pages 149–158, 2008.

[Fagin et al., 2005] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. Theor. Comput. Sci., 336(1):89–124, 2005.

[Gogacz and Marcinkowski, 2017] Tomasz Gogacz and Jerzy Marcinkowski. Converging to the chase - A tool for finite controllability. J. Comput. Syst. Sci., 83(1):180–206, 2017.

[Gottlob et al., 2013] Georg Gottlob, Marco Manna, and Andreas Pieris. Combining decidability paradigms for existential rules. Theory and Practice of Logic Programming, 16(1):877–892, 2013.

[Grau et al., 2013] Bernardo Cuenca Grau, Ian Horrocks, Markus Krötzsch, Clemens Kupke, Despoina Magka, Boris Motik, and Zhe Wang. acyclicity notions for existential rules and their application to query answering in ontologies. Journal of Artificial Intelligence Research, 47:741–808, 2013.

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

[Krötzsch and Rudolph, 2011] Markus Krötzsch and Sebastian Rudolph. Extending decidable existential rules by joining acyclicity and guardedness. In IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011, pages 963–968, 2011.

[Leone et al., 2012] Nicola Leone, Marco Manna, Giorgio Terracina, and Pierfrancesco Veltri. Efficiently computable datalog-\exists programs. In Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012, 2012.

[Maier et al., 1979] David Maier, Alberto O. Mendelzon, and Yehoshua Sagiv. Testing implications of data dependencies. ACM Trans. Database Syst., 4(4):455–469, 1979.

[Patel-Schneider and Horrocks, 2007] Peter F. Patel-Schneider and Ian Horrocks. A comparison of two modelling paradigms in the semantic web. J. Web Sem., 5(4):240–250, 2007.

[Rosati, 2006] Riccardo Rosati. On the decidability and finite controllability of query processing in databases with incomplete information. In Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 26-28, 2006, Chicago, Illinois, USA, pages 356–365, 2006.

[Rosati, 2011] Riccardo Rosati. On the finite controllability of conjunctive query answering in databases under open-
world assumption. *J. Comput. Syst. Sci.*, 77(3):572–594, 2011.

[Zhang et al., 2015] Heng Zhang, Yan Zhang, and Jia-Huai You. Existential rule languages with finite chase: Complexity and expressiveness. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI-2015)*, pages 1678–1685, 2015.