Static analysis of Frontier Guarded Existential Rules

Paul RAPHAEL

February-August 2026

1 Introduction

Databases represent information by storing facts as atoms, typically under the *closed-world assumption*: any fact not present in the database is considered false. In many settings, however, one may have additional knowledge about the world that is not explicitly stored in the database. To reason about such implicit knowledge, one can extend the database with a *logical layer*, forming a *knowledge base*. In this context, a fact is considered true if it is entailed by the combination of the database and the logical layer. Users can then pose queries over the knowledge base to retrieve information that may not be explicitly stored.

In this paper, we focus on a class of logical layers known as *tuple-generating dependencies (TGDs)*, or *existential rules*. Within this framework, several subclasses of rules have been studied, including *linear rules* (where the body is atomic), *guarded rules*, and *frontier-guarded rules*, which impose various syntactic restrictions on how variables appear in rule bodies and heads.

A central type of query over knowledge bases is the *conjunctive query (CQ)*, which consists of a conjunction of atoms. A fundamental problem is to determine whether a given CQ is entailed by a knowledge base. Since the logical layer may allow infinitely many rule applications, the canonical model of a knowledge base may itself be infinite. To address this issue, one may try to *rewrite* the query into an equivalent query that can be evaluated directly over the original database. While it is well known that every conjunctive query is rewritable under linear TGDs [1], this property does not hold in general for guarded or frontier-guarded TGDs.

Another fundamental question concerns the behavior of the *chase* procedure. Indeed, applying a TGD may introduce new atoms, which in turn enable further rule applications. As a consequence, the chase may generate an infinite sequence of atoms and never terminate. Determining whether the chase terminates has therefore been extensively studied [2]. Chase termination and query rewritability are closely related: both seek finite representations of the potentially infinite consequences of a knowledge base, although they do so in different ways.

The main result of this paper is that, for guarded TGDs, the rewritability of arbitrary conjunctive queries is equivalent to the rewritability of atomic queries. In other words, deciding whether every conjunctive query admits a finite rewriting can be reduced to deciding this property for atomic queries alone. We also investigate the frontier-guarded setting, where we obtain a different characterization. Finally, we study the problem of chase termination for guarded and frontier-guarded TGDs.

The remainder of this report is organized as follows. Section 2 introduces the necessary preliminaries on tuple-generating dependencies, conjunctive queries, ontology-mediated queries, and the semi-oblivious chase. Section 3 formally introduces the two static-analysis problems studied in this report: uniform first-order rewritability and uniform termination of the semi-oblivious chase. Section 4 presents the linearization of guarded TGDs into linear TGDs, which will be used in the study of uniform rewritability. Section 5 establishes the complexity of uniform first-order rewritability for guarded TGDs, by proving both the upper and lower bounds. Section 6 investigates to what extent the techniques developed for guarded TGDs can be extended to frontier-guarded TGDs, and presents the corresponding characterization in terms of sub-body queries. Section 7 introduces a linearization procedure adapted to frontier-guarded TGDs. Finally, Section 8 uses this linearization to establish the complexity of uniform termination of the semi-oblivious chase for frontier-guarded TGDs, before we conclude in Section 9.

2 Preliminaries

Relational Structures. We assume three pairwise disjoint countably infinite sets of constants, nulls, and variables, denoted by C, N, and V, respectively. A *term* is either a constant, a null, or a variable. For an integer $n > 0$, we write $[n] = \{1, \ldots, n\}$.

A *relational schema* is a finite set of relation symbols, each associated with a non-negative integer called its *arity*. We write R/n to denote that the relation symbol R has arity n, and write $\mathrm{ar}(R) = n$ when convenient.

An *atom* over a schema is an expression of the form

$$R(t_1, \ldots, t_n),$$

where R/n belongs to the schema and each t_i is a term.

Instances and Homomorphisms. An *instance* is a (possibly infinite) set of atoms containing only constants and nulls. A *database* is a finite instance containing only constants. The *active domain* of an instance I, denoted $\mathrm{dom}(I)$, is the set of terms occurring in I.

Given two sets of atoms A and B, a *homomorphism* from A to B is a mapping h from the terms occurring in A to the terms occurring in B that fixes every constant and satisfies

$$R(h(\bar{t})) \in B$$

for every atom $R(\bar{t}) \in A$, where $h(\bar{t})$ denotes the tuple obtained by applying h componentwise to $\bar{t}$. We write

$$A \to B$$

whenever there exists a homomorphism from A to B.

Tuple-Generating Dependencies. A *tuple-generating dependency* (TGD) is a constant-free first-order sentence of the form

$$\forall \bar{x} \forall \bar{y} \big(\phi(\bar{x}, \bar{y}) \to \exists \bar{z}\, \psi(\bar{x}, \bar{z}) \big),$$

where $\phi(\bar{x}, \bar{y})$ and $\psi(\bar{x}, \bar{z})$ are non-empty conjunctions of atoms. As usual, we omit universal quantifiers and write

$$\phi(\bar{x}, \bar{y}) \to \exists \bar{z}\, \psi(\bar{x}, \bar{z}).$$

The conjunctions $\phi(\bar{x}, \bar{y})$ and $\psi(\bar{x}, \bar{z})$ are called the *body* and the *head* of the TGD, and are denoted by $\mathrm{body}(\sigma)$ and $\mathrm{head}(\sigma)$, respectively. The variables occurring in both the body and the head are called the *frontier variables* of σ and are denoted by $\mathrm{fr}(\sigma)$.

An instance I satisfies a TGD σ, written $I \models \sigma$, if every homomorphism from $\mathrm{body}(\sigma)$ to I extends to a homomorphism from $\mathrm{head}(\sigma)$ to I. A set of TGDs Σ is satisfied by I, written $I \models \Sigma$, if $I \models \sigma$ for every $\sigma \in \Sigma$.

Guarded and Frontier-Guarded TGDs. A TGD σ is *guarded* if there exists an atom in $\mathrm{body}(\sigma)$ containing every variable occurring in the body of σ. Such an atom is called a *guard* of σ.

Similarly, σ is *frontier-guarded* if there exists an atom in $\mathrm{body}(\sigma)$ containing every frontier variable of σ. Such an atom is called a *frontier guard*. Every guarded TGD is frontier-guarded, but the converse does not hold.

Conjunctive Queries. A *Boolean conjunctive query* (CQ) is a first-order sentence of the form

$$Q = \exists \bar{x} \left(A_1 \wedge \cdots \wedge A_n \right),$$

where each A_i is an atom whose variables belong to $\bar{x}$.

We identify Q with the set of atoms $\{A_1, \ldots, A_n\}$. An instance I satisfies Q, written $I \models Q$, if there exists a homomorphism from Q to I.

Ontology-Mediated Queries. An *ontology-mediated query* (OMQ) is a triple

$$O = (S, \Sigma, Q),$$

where S is a relational schema, Σ is a finite set of TGDs, and Q is a conjunctive query over the schema obtained by extending S with the relation symbols occurring in Σ.

Given an S-database D, a *model* of (D, Σ) is an instance $I \supseteq D$ such that $I \models \Sigma$. The answer to O over D is the set

$$\mathrm{ans}(O, D) = \bigcap_{I \models (D, \Sigma)} Q(I),$$

called the *certain answers* of O over D.

The Semi-Oblivious Chase. The semi-oblivious chase is a procedure that constructs a universal model of a database and a set of TGDs by repeatedly applying rules.

A *trigger* for a TGD σ on an instance I is a homomorphism from $\mathrm{body}(\sigma)$ to I. Applying a trigger extends I with a fresh copy of $\mathrm{head}(\sigma)$, where frontier variables are mapped according to the trigger and each existentially quantified variable is replaced by a fresh null determined by the trigger.

Starting from a database D, the semi-oblivious chase repeatedly applies active triggers until no active trigger remains, or indefinitely if this process does not terminate. The resulting instance is denoted by

$$\mathrm{chase}(D, \Sigma).$$

It is well known that the semi-oblivious chase is universal, in the sense that it homomorphically maps to every model of (D, Σ). Consequently, conjunctive queries can be evaluated directly on the chase, as proved in [3]

Proposition 2.1. *For every ontology-mediated query $O = (S, \Sigma, Q)$ and every S-database D,*

$$\mathrm{ans}(O, D) = Q(\mathrm{chase}(D, \Sigma)).$$

For every integer $k \geq 0$, we denote by $\mathrm{chase}_k(D, \Sigma)$ the instance obtained after at most k rounds of the semi-oblivious chase. Thus,

$$\mathrm{chase}_0(D, \Sigma) = D,$$

and, for every $k \geq 0$, $\mathrm{chase}_{k+1}(D, \Sigma)$ is obtained from $\mathrm{chase}_k(D, \Sigma)$ by applying all active triggers to $\mathrm{chase}_k(D, \Sigma)$. The result of the semi-oblivious chase is then

$$\mathrm{chase}(D, \Sigma) = \bigcup_{k \geq 0} \mathrm{chase}_k(D, \Sigma).$$

3 Static Analysis of TGD's

We now introduce the two main decision problems considered in this study. Briefly, given a set of TGDs Σ, we are interested in deciding whether:

1. every conjunctive query admits a finite first-order rewriting;
2. the chase terminates for every database.

Let us now give more detail on each problem.

3.1 Uniform First-Order Rewritability

The goal of query rewriting is to answer queries over a knowledge base without explicitly computing its consequences under the TGDs. Instead, one seeks to transform the original query into an equivalent first-order query that can be evaluated directly over the input database. Uniform first-order rewritability asks whether such a rewriting exists simultaneously for every conjunctive query over a fixed data schema.

We now formalize this notion.

Definition 3.1 (Uniform First-Order Rewritability)**.** *Let Σ be a set of TGDs over a schema S. We say that Σ is* uniformly first-order rewritable *relative to S if, for every conjunctive query Q over S, there exists a UCQ query Q' such that, for every database D over S,*

$$Q'(D) = \mathrm{cert}(Q, D, \Sigma).$$

This naturally gives rise to the following decision problem. Given the class of TGDs C, we define:

Problem:	$\mathsf{UniformRewritability(C)}$
Input:	A set $\Sigma \in C$ of TGDs and a data schema S.
Question:	Is Σ uniformly first-order rewritable relative to S?

The complexity of this problem will be our main focus for guarded and frontier-guarded TGDs.

Theorem 3.2. *The problem* $\mathsf{UniformRewritability}$ *is 2-ExpTime-complete for both guarded and frontier-guarded TGDs.*

We first focus on the guarded case.

The proof of Theorem 3.2 is given in Section 5. The upper bound is based on a characterization showing that uniform first-order rewritability can be reduced to the rewritability of a restricted family of atomic queries. More precisely, it is sufficient to consider ground atomic queries over the signature induced by the data schema and the predicates occurring in Σ. Since there are only exponentially many such queries (up to renaming of constants), deciding uniform rewritability amounts to checking first-order rewritability for each of them individually. As first-order rewritability of atomic queries can be decided in 2-ExpTime, this yields the desired upper bound.

The key ingredient of this characterization is a linearization procedure, presented in Section 4, which transforms a set of guarded TGDs into an equivalent set of simple-linear TGDs for query answering. This construction is inspired from [4]

3.2 Uniform Termination of the Semi-Oblivious Chase

Another central question is whether the semi-oblivious chase always terminates. More precisely, given a set of TGDs Σ, we ask whether the chase terminates on every database over a fixed data schema. When this is the case, the chase can always be computed in finite time, making it possible to answer conjunctive queries by evaluating them directly over the chase instance.

We now formalize this notion.

Definition 3.3 (Uniform Semi-Oblivious Chase Termination). *Let Σ be a set of TGDs and let S be a data schema. We say that Σ is* uniformly terminating *relative to S if, for every database D over S, the semi-oblivious chase $Chase(D, \Sigma)$ is finite.*

This leads to the following decision problem.

Problem:	UniformTermination
Input:	A set Σ of TGDs and a data schema S.
Question:	Does the semi-oblivious chase terminate on every database over S?

For guarded TGDs, the complexity of UniformTermination is already known [5]. In this paper, we focus on the frontier-guarded setting and establish the corresponding complexity result.

Theorem 3.4. *The problem* UniformTermination *is* 2-EXPTIME*-complete for frontier-guarded TGDs.*

4 Linerization of Guarded TGDs

Linearisation is a transformation that reduces query answering under guarded TGDs to query answering under linear TGDs by explicitly encoding local consequences of the rules into the database.

Intuition. Given a database D and a set of TGDs Σ, the completion $D_\Sigma = complete(D, \Sigma)$ contains all ground facts over $\mathrm{dom}(D)$ that are entailed by Σ. The key idea of linearisation is to *compress* the information in D_Σ by grouping, for each atom, all facts that can be derived over the same domain.

More precisely, instead of representing each fact individually, we associate to every atom $\alpha = R(\bar{a})$ its Σ-type, which consists of α together with all facts in D_Σ whose terms are drawn from $\mathrm{dom}(\alpha)$. This local neighbourhood is then represented by a single fresh predicate symbol $[\tau]$, called a *type atom.*

Σ-types Let Σ be a set of TGDs over a schema $\mathcal{S}$. Let $\alpha = R(\bar{t})$ be an atom.

Two tuples $\bar{t}$ and $\bar{u}$ of the same length are said to have the same *equality pattern*, written $\bar{t} \simeq \bar{u}$, if for all positions i, j it holds that $t_i = t_j$ iff $u_i = u_j$.

A Σ-*type* is a pair $\tau = (\gamma, T)$ where:

- γ is an atom $R(\bar{u})$ whose terms are taken from $\{1, \ldots, k\}$ for some $k \leq \mathrm{ar}(\Sigma)$, and
- T is a set of atoms such that $\mathrm{dom}(T) \subseteq \mathrm{dom}(\gamma)$ and $\gamma \notin T$.

Intuitively, γ represents a canonical guard atom together with its equality pattern, while T represents all additional atoms over the same domain that are entailed under Σ.

Given an atom $\alpha = R(\bar{t})$, we say that a Σ-type $\tau = (\gamma, T)$ belongs to Σ-types(α) if $\gamma = R(\bar{u})$ and $\bar{t} \simeq \bar{u}$.

For a database D, the Σ-*type of* α *in* D is the type

$$type_{D,\Sigma}(\alpha) = (\gamma, T)$$

such that γ matches the equality pattern of α and

$$T = \{\beta \in complete(D, \Sigma) \mid \mathrm{dom}(\beta) \subseteq \mathrm{dom}(\alpha)\} \setminus \{\alpha\}.$$

Linearised database. The linearisation of D, denoted $lin(D)$, is obtained by replacing each atom $\alpha = R(\bar{a})$ of D with a single atom $[\tau](\bar{a})$, where $\tau = type_{D,\Sigma}(\alpha)$. Thus, each atom of $lin(D)$ compactly represents all information about α that is derivable under Σ within its domain.

Linearised rules. Each guarded TGD is transformed into a set of linear TGDs. Intuitively, whenever a rule can be applied to a type τ, we produce new type atoms corresponding to the Σ-types of the atoms in the head of the rule. In this way, the application of rules is simulated at the level of types, and every rule has a single atom in its body.

Projection to the original schema. Each type atom represents a collection of atoms over the original schema. To recover these atoms, we introduce, for every Σ-type $\tau = (\gamma, T)$ and every atom $\beta \in \{\gamma\} \cup T$, a linear rule

$$[\tau](\bar{x}) \to \beta(\bar{x}),$$

where the variables of β are instantiated according to the canonical ordering of the terms of τ. These projection rules ensure that every fact encoded by a type atom can be recovered over the original schema, allowing conjunctive queries over the original predicates to be evaluated without modification.

Theorem 4.1 (Correctness of the guarded linearisation)**.** *Let D be a database over the original schema S, and let Σ be a finite set of guarded TGDs. Let*

$$I = \mathrm{chase}(D, \Sigma) \qquad \textit{and} \qquad J = \mathrm{chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma)).$$

Denote by $J|_S$ the restriction of J to predicates of the original schema S. Then I and $J|_S$ are homomorphically equivalent:

$$I \to J|_S \qquad \textit{and} \qquad J|_S \to I.$$

Consequently, for every conjunctive query Q over S,

$$I \models Q \iff J \models Q.$$

Proof. We prove the two homomorphisms separately.

From the original chase to the linearised chase. We construct, by induction on the chase derivation of I, a mapping

$$h : I \to J|_S.$$

Initially, every atom $\alpha = R(\bar{a})$ of D is replaced in $\mathrm{Lin}(D)$ by a type atom

$$[\tau](\bar{a}),$$

where τ is the type of α. By definition, τ contains α. Hence, the decoding rule associated with α derives

$$R(\bar{a})$$

in J. Therefore, the identity mapping on the constants of D defines a homomorphism from D to $J|_S$.

Assume now that the mapping has been defined after some finite number of chase steps. Consider an application of a rule

$$\sigma : \phi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z}\, \psi(\bar{x}, \bar{z})$$

in the original chase. Let g be the corresponding trigger, so that

$$g(\phi) \subseteq I.$$

Since σ is guarded, its body contains an atom γ whose variables include all variables of ϕ. The image $g(\gamma)$ therefore contains all terms used by the trigger. By the induction hypothesis, the image of the body under $h \circ g$ occurs in $J|_S$. These local facts are encoded by a type atom associated with $h(g(\gamma))$.

The construction of $\mathrm{Lin}(\Sigma)$ contains a linear rule simulating this application of σ on that type. Applying this rule creates type atoms for the head atoms of σ. The decoding rules then produce the corresponding original atoms in $J|_S$.

Map every fresh null introduced by the original chase step to the fresh null introduced by the corresponding linearised chase step. Extending h in this way preserves all newly generated head atoms. Thus, the induction invariant is maintained.

Taking the union of these compatible finite mappings yields a homomorphism

$$h : I \rightarrow J|_S.$$

From the linearised chase to the original chase. We now construct a homomorphism

$$k : J|_S \rightarrow I$$

by induction on the derivation of type atoms in the linearised chase.

Consider first a type atom occurring in the linearised database:

$$[\tau](\bar{a}) \in \mathrm{Lin}(D).$$

By definition of $\mathrm{Lin}(D)$, the type τ records atoms that are entailed from D and Σ over the domain of the corresponding guard atom. Since the chase is a universal model of (D, Σ), every original-schema atom decoded from $[\tau](\bar{a})$ occurs in I. Hence, the identity on $\bar{a}$ maps all such decoded atoms to I.

Suppose now that a type atom

$$[\tau'](\bar{b})$$

is generated by a linearised rule from a previously generated type atom

$$[\tau](\bar{a}).$$

By construction of $\mathrm{Lin}(\Sigma)$, this linear rule exists only if there is a rule $\sigma \in \Sigma$ and a homomorphism from the body of σ into the collection of atoms encoded by τ. Moreover, τ' describes the atoms obtained by applying σ under this homomorphism and by closing the resulting local instance under Σ.

By the induction hypothesis, all atoms encoded by τ map to I. Thus, the corresponding body homomorphism gives a valid trigger for σ in I. Since $I = \mathrm{chase}(D, \Sigma)$ is closed under applications of rules from Σ, the required head atoms are present in I. We map the fresh nulls introduced in the linearised step to witnesses produced by the corresponding rule application in I. Every original-schema atom decoded from $[\tau'](\bar{b})$ is therefore preserved.

Taking the union of these mappings over the linearised chase derivation gives a homomorphism

$$k : J|_S \rightarrow I.$$

We have proved that I and $J|_S$ are homomorphically equivalent. Since conjunctive queries are invariant under homomorphic equivalence, for every conjunctive query Q over S,

$$I \models Q \iff J|_S \models Q.$$

As Q uses only predicates from S, this is equivalent to

$$I \models Q \iff J \models Q.$$

□

5 Uniform First-Order Rewritability of Guarded TGDs

We are now ready to discuss the proof of Theorem 3.2. We start with the upper bound in Section 5.1 and then establish the lower bound in Section 5.2.

5.1 Upper Bound

5.1.1 Rewriting over Complete Databases

The proof of the upper bound begins with a rewriting result for complete databases. More precisely, we first show that, once a database contains all consequences relevant to its active domain, conjunctive-query answering under guarded TGDs can be reduced to the evaluation of a union of conjunctive queries over that database.

This first step is formalized by the following claim:

Claim 5.1. *Given Σ a set of Guarded TGDs and a CQ Q, there is a ucq Q_Σ such that, for every database D,*

$$D_\Sigma \wedge \Sigma \models Q \iff D_\Sigma \models Q_\Sigma$$

Note that thank to theorem 4.1, we have the following proposition:

Proposition 5.2. *Given a set of guarded TGD Σ and a CQ Q, for every database D,*

$$D_\Sigma \wedge \Sigma \models Q \iff lin(D) \wedge lin(D_\Sigma) \models Q$$

The difficulty is that guarded TGDs do not enjoy the same rewriting properties as linear TGDs. The idea is therefore to transform the problem into one involving linear TGDs. Using the linearization procedure, we replace the original ontology by an equivalent linear one, compute a UCQ rewriting in this simpler setting, and finally show that this rewriting correctly answers the original query over the completed database.

Definition 5.3 (Delinearisation of a UCQ over a linearised database)**.** *Let $Q' = \bigvee_{i=1}^n C_i$ be a UCQ over the linearised schema, where each C_i is a conjunctive query consisting of linearised atoms $[\tau](\bar{x})$. The* delinearisation *of Q', denoted Q'', is the UCQ over the original schema obtained by replacing each linearised atom $[\tau](\bar{x})$ with the set of atoms $\tau(\bar{x})$ it represents:*

$$Q'' = \bigvee_{i=1}^{n} \left(\bigwedge_{[\tau](\bar{x}) \in C_i} \tau(\bar{x}) \right),$$

where for each τ, $\tau(\bar{x})$ is the set of original atoms associated with the Σ-type τ.

Proof of Claim 5.1. We know that there exist a UCQ Q^l such that for every database D $lin(D) \wedge lin(\Sigma) \models lin(Q) \iff lin(D) \models Q^l$. We know want to prove that

$$lin(D) \models Q^l \iff D_\Sigma \models delin(Q^l)$$

which prove Claim 5.1.

We prove both implications.

($\Rightarrow$) Assume $lin(D) \models Q^l$. Since Q^l is a UCQ, there exists a disjunct q^l of Q^l and a homomorphism

$$h : q^l \to lin(D).$$

Every atom of q^l is of the form $[\tau](\bar{x})$ for some Σ-type τ. By construction of $lin(D)$, we have:

$$lin(D) \models [\tau](\bar{a}) \quad \text{iff} \quad D_\Sigma \models \tau(\bar{a}).$$

Hence, for every atom $[\tau](\bar{x})$ of q^l,

$$D_\Sigma \models \tau(h(\bar{x})).$$

Let q be the CQ obtained from q^l by replacing each atom $[\tau](\bar{x})$ with the conjunction $\tau(\bar{x})$. Then the same homomorphism h witnesses that

$$D_\Sigma \models q.$$

Since q is a disjunct of $\mathrm{delin}(Q^l)$, we conclude that

$$D_\Sigma \models \mathrm{delin}(Q^l).$$

($\Leftarrow$) Assume $D_\Sigma \models \mathrm{delin}(Q^l)$. Then there exists a disjunct q of $\mathrm{delin}(Q^l)$ and a homomorphism

$$h : q \to D_\Sigma.$$

Each atom of q belongs to some Σ-type $\tau(\bar{x})$. Since $D_\Sigma \models \tau(h(\bar{x}))$, by construction of $lin(D)$ we obtain

$$lin(D) \models [\tau](h(\bar{x})).$$

Let q^l be the CQ obtained by replacing each $\tau(\bar{x})$ in q with $[\tau](\bar{x})$. Then the same homomorphism h satisfies

$$lin(D) \models q^l.$$

Since q^l is a disjunct of Q^l, we conclude that

$$lin(D) \models Q^l.$$

Therefore,

$$lin(D) \models Q^l \quad \iff \quad D_\Sigma \models \mathrm{delin}(Q^l).$$

□

5.1.2 Characterization of Uniform Rewritability

We now show that the rewriting result established in the previous subsection completely characterizes uniform first-order rewritability. The forward implication is immediate, since atomic conjunctive queries are a particular case of conjunctive queries. The converse relies on the fact that every conjunctive query over a complete database can be rewritten as a union of conjunctive queries, and that each ground atomic query admits a first-order rewriting by assumption.

Theorem 5.4. *Let Σ be a set of guarded TGDs. The following are equivalent:*

1. *(Σ, CQ) is first-order rewritable;*
2. *(Σ, ACQ) is first-order rewritable.*

Proof. The implication (1) $\Rightarrow$ (2) is immediate.

Assume now that every atomic conjunctive query is first-order rewritable with respect to Σ. Let Q be an arbitrary conjunctive query.

By Claim 5.1, there exists a UCQ Q_Σ such that, for every database D,

$$D, \Sigma \models Q \qquad \Longleftrightarrow \qquad D_\Sigma \models Q_\Sigma,$$

where D_Σ denotes the completion of D with respect to Σ.

Since Q_Σ is a union of conjunctive queries, it is enough to rewrite each disjunct independently. Hence, assume that

$$Q_\Sigma = \bigwedge_{i=1}^{m} \alpha_i,$$

where each α_i is an atomic formula.

The atoms of Q_Σ may share variables. To account for every possible identification between these variables, we consider all partitions of the variables of Q_Σ.

Let π be such a partition. To every equivalence class of π, we associate a fresh constant. Replacing each variable by the constant associated with its class yields a ground conjunction Q_Σ^π.

By assumption, every ground atom occurring in Q_Σ^π admits a first-order rewriting. Rewriting each atom independently and taking their conjunction yields a UCQ φ_π satisfying

$$D, \Sigma \models Q_\Sigma^\pi \qquad \Longleftrightarrow \qquad D \models \varphi_\pi.$$

The rewriting procedure for ground atomic queries is invariant under renaming of the fresh constants. Therefore, replacing each fresh constant by the variable representing its equivalence class gives a UCQ $\widehat{\varphi}_\pi$.

Finally, define

$$\Phi = \bigvee_{\pi} \widehat{\varphi}_\pi,$$

where the disjunction ranges over all partitions of the variables of Q_Σ.

By construction, every homomorphism from Q_Σ into a complete database induces exactly one partition of its variables, and conversely every satisfying assignment of one disjunct of Φ defines a homomorphism from Q_Σ. Hence,

$$D_\Sigma \models Q_\Sigma \qquad \Longleftrightarrow \qquad D \models \Phi.$$

Combining this equivalence with Claim 5.1, we obtain

$$D, \Sigma \models Q \qquad \Longleftrightarrow \qquad D \models \Phi,$$

showing that Q is first-order rewritable with respect to Σ. $\square$

5.1.3 Decision Procedure and Complexity Analysis

We now use Theorem 5.4 to derive the upper bound for uniform first-order rewritability. The characterization established in the previous subsection shows that it is sufficient to decide whether all atomic queries are first-order rewritable with respect to the input set of guarded TGDs.

We rely on the following known result, proved in [6]

Theorem 5.5 ([6])**.** *Given a set Σ of guarded TGDs, a data schema S, and a ground atomic query q, deciding whether q is first-order rewritable with respect to Σ over S can be done in* 2-ExpTime.

We can now establish the desired upper bound.

Theorem 5.6. *The problem* UniformRewritability(G) *is in* 2-ExpTime.

Proof. Let Σ be a finite set of guarded TGDs and let S be the data schema. By Theorem 5.4, Σ is uniformly first-order rewritable relative to S if and only if every atomic query over

$$S \cup \mathrm{sch}(\Sigma)$$

is first-order rewritable with respect to Σ.

It is enough to consider ground atomic queries up to renaming of constants. Indeed, for a predicate R of arity n, a ground atomic query

$$R(c_1, \ldots, c_n)$$

is determined, up to constant renaming, by the equality pattern of the tuple $(c_1, \ldots, c_n)$. Such an equality pattern is represented by a partition of the set $[n]$. We may therefore choose one canonical ground atom for each predicate and each equality pattern of its arguments.

The number of partitions of $[n]$ is the n-th Bell number, which is bounded by

$$2^{O(n \log n)}.$$

Consequently, the number of canonical ground atomic queries over $S \cup \mathrm{sch}(\Sigma)$ is at most exponential in the size of the input.

The decision procedure is therefore the following:

1. enumerate all canonical ground atomic queries over $S \cup \mathrm{sch}(\Sigma)$;
2. for each such query q, apply the decision procedure of Theorem 5.5;
3. accept if and only if every query considered is first-order rewritable with respect to Σ.

The correctness of the procedure follows directly from Theorem 5.4. Indeed, the procedure accepts precisely when all atomic queries are first-order rewritable, which is equivalent to uniform first-order rewritability for conjunctive queries.

It remains to analyze its running time. There are at most exponentially many canonical ground atomic queries, and each invocation of the atomic rewritability procedure takes double-exponential time. Thus, the complete procedure runs in 2-ExpTime, proving the claim. □

5.2 Lower Bound

We introduce the following problem, where D is a fixed database composed of only one atom, so we will confuse the database and the atom,

Problem :	QueryAnswering(D)
Input :	A set Σ of guarded TGDs, an atomic boolean query q without constant.
Question :	Does the chase of D and Σ entail q ?

In [7], it is proven that

- This problem is 2-EXPTIME-complete when the arity of the predicate is not bounded

- This problem is EXPTIME-complete when the arity of the predicate is bounded

We can assume all rule heads are atomic. To prove Theorem 3.2, we will reduce this problem to UniformRewritability(Guarded)

Σ be a set of guarded TGDs with atomic heads, and q be a boolean atomic query.

We will now define a few set of rules. We assume that all the introduced predicates are fresh, meaning they dont appear in Σ or D.

- $\Sigma_p = \{p(x) \wedge r(x,y) \rightarrow p(y), r(x,y) \rightarrow \exists \bar{z} \text{init}(x, \bar{z}), \text{init}(x, \bar{z}) \rightarrow D(\bar{z})\}$ where $\bar{z}$ has the arity of D.
- $\Sigma_{\text{flood}} = \{\text{flood}(x) \rightarrow h(x, \ldots, x), q(x_1, \ldots, x_n) \rightarrow \text{flood}(x_1)\}$ for all predicate h.

Note that both those rule set are guarded.

We now define $\Sigma' = \Sigma_p \cup \mathcal{R}_{\text{flood}} \cup \Sigma$.

Proposition 5.7. *Σ' is uniformly rewritable if and only if the chase of D and Σ entails q.*

Proof. We will show both sides of the implication.

We prove the forward implication by contraposition. For this we show that if the chase of D and Σ does not entail q, then Σ' is not uniformly rewritable. Indeed, query $p(x) \wedge a(x)$t is not rewritable. Indeed consider database $D_n = \{p(x_1), r(x_1, a_2), \ldots r(a_{n-1}, b), a(b)\}$ that has schema S. Since the query q is not in the chase of D and Σ, then the flood is never derived. Hence, in Σ', the only way to derive an atom of the form $p(x)$ is to use the rule $p(x) \wedge r(x,y) \rightarrow p(y)$. On database D_n, that rule has to be applied at least n times in succession to derive $p(a_n)$. Indeed, we have

$$\text{Chase}_1(D_n, \Sigma')_{|S} = (a(b), p(x_1), p_(x_2), r(x_1, a_2), \ldots r(a_{n-1}, b),$$

then

$$\text{Chase}_2(D_n, \Sigma')_{|S} = (a(b), p(x_1), p_(x_2), p(x_3), r(x_1, a_2), \ldots r(a_{n-1}, b),$$

ect. Hence, since we can take n as large as we want and since this is the only way to derive $p(b)$, then we know that query $p(a) \wedge p(b)$ is not rewritable because we do not respect the BDDP, hence the set of TGDs Σ' is not uniformly rewritable.

Let us now show that if q is entailed by Σ and D, then Σ' is uniformly rewritable. For the rewritability, we work on schema $\{p, r\}$. Let Q be a boolean query with no constant. Let D' be a database on S. If there is an atom of form $r(x,y)$, then by deriving the atom D, we can then derive an atom of the from $q(x_1, \ldots, x_n)$, that then allows us to derive the flood to retrieve flood(z). Hence, we can for any atom of the form $r(x,y)$, derive flood(z). Then, with the rule flood$(z) \rightarrow h(z, \ldots, z)$, we can satisfy any query. If D' has no atom of form $r(x,y)$ then no rule can fire, so if there are only atoms of form $p(x)$ then Q is a rewriting of itself. Hence, a rewriting of the query can always be $r(x,y) \vee Q$, so Q is indeed first order rewritable. Hence, $\mathcal{R}'$ is uniformably rewritable ☐

Note that the creation of rules is linear in Σ. This claim allows to prove both point of Theorem 3.2:

Proof of Theorem 3.2. We will prove both points seperately.

For the problem UniformRewritability(Guarded) with unbounded predicate, we have a clear reduction from QueryAnswering(D) with unbounded arity. We know that problem is 2-EXPTIME-complete. The reduction we presented runs in polynomial time, and thanks to Proposition 5.7 we know that our reduction holds.

For the problem UniformRewritability(Guarded) with bounded predicate, we do the same reduction, and note that the rules created also have bounded arity, since the newly created predicate have arity smaller than the arity of the database. ☐

Note that since we have the reduction for guarded we also have it for frontier guarded.

6 Towards Frontier-Guarded TGDs

The proof developed in the previous section suggests a natural question: how much of the argument actually depends on guardedness?

The key ingredient is Claim 5.1. Recall that, for every set Σ of guarded TGDs and every conjunctive query Q, there exists a UCQ Q_Σ such that, for every database D,

$$D_\Sigma, \Sigma \models Q \quad \Longleftrightarrow \quad D_\Sigma \models Q_\Sigma.$$

In other words, once the input database has been completed with respect to Σ, conjunctive-query answering under Σ becomes UCQ-rewritable.

The remainder of the proof of uniform rewritability relies only on this property. Indeed, once such a UCQ Q_Σ is available, the argument of Section 5.1 reduces its atoms to atomic queries and uses their first-order rewritings. This naturally raises the following question:

Does Claim 5.1 remain true for frontier-guarded TGDs?

If the answer were positive, the same proof strategy could be reused almost unchanged in the frontier-guarded setting. Unfortunately, this is not the case. Even over complete databases, conjunctive queries under frontier-guarded TGDs are not necessarily UCQ-rewritable.

The following example shows that our proprety does not hold for frontier guarded TGDs.

Example 6.1. *Consider the following frontier-guarded TGD:*

$$\Sigma = \{s(x, y),\, r(x, z) \to \exists t\, s(z, t)\}\,.$$

This rule is frontier-guarded: the only frontier variable is z, and it is guarded by the atom $r(x, z)$. Notice, however, that the rule is not guarded, since no atom in its body contains all three variables x, y, z.

Consider the Boolean conjunctive query

$$Q = \exists x\, \exists y\, \big(q(x) \wedge s(x, y)\big).$$

For every $n \geq 1$, let

$$D_n = \{p(a_1), r(a_1, a_2), r(a_2, a_3), \ldots, r(a_{n-1}, a_n), q(a_n), s(a_1, b_1)\}.$$

Starting from $s(a_1, b_1)$, the rule can propagate an s-atom along the r-path. More precisely, after one application we obtain

$$s(a_2, b_2),$$

where b_2 is a fresh null; after two applications we obtain

$$s(a_3, b_3),$$

and so on. Eventually, after $n - 1$ successive applications, the chase contains

$$s(a_n, b_n).$$

Since $q(a_n) \in D_n$, it follows that

$$\mathrm{chase}(D_n, \Sigma) \models Q.$$

On the other hand, the rule never creates a new ground atom: every generated s-atom contains a fresh null in its second position. Consequently, completing D_n with respect to ground consequences does not reveal the propagation along the r-path.

Moreover, the number of chase rounds needed to satisfy Q is unbounded: for the database D_n, at least $n-1$ successive rule applications are required before an atom of the form $s(a_n, t)$ is generated. Hence there is no uniform bound k, independent of the database, such that

$$\mathrm{chase}(D, \Sigma) \models Q \quad \Longrightarrow \quad \mathrm{chase}_k(D, \Sigma) \models Q.$$

By the bounded derivation depth characterization of UCQ rewritability, Q is therefore not UCQ-rewritable with respect to Σ as proved in [1]

Thus, even though Σ is frontier-guarded, conjunctive-query answering over complete databases is not necessarily UCQ-rewritable. This shows that the characterization that holds for guarded TGDs, does not extend directly to frontier-guarded TGDs.

Although our characterization does not extend directly to frontier-guarded TGDs, this does not mean that uniform first-order rewritability cannot be characterized for this larger class. Instead, a different family of queries has to be considered.

For this purpose, we introduce the notion of *sub-body queries*. Intuitively, these queries are obtained by selecting some atoms from the body of a rule and existentially quantifying all variables occurring in the selected atoms.

Definition 6.2 (Sub-body Queries)**.** *Let Σ be a set of TGDs. We define the set of* sub-body queries *of Σ, denoted $\mathcal{B}(\Sigma)$, as*

$$\mathcal{B}(\Sigma) = \left\{ \exists \bar{z} \bigwedge_{\beta \in A} \beta(\bar{x}) \;\middle|\; \sigma \in \Sigma,\ A \subseteq \mathrm{body}(\sigma),\ \bar{x} = \mathrm{terms}(A),\ \bar{z} = \mathrm{terms}(A) \setminus \mathcal{C} \right\},$$

where $\mathcal{C}$ denotes the set of constants.

In other words, a sub-body query is obtained from an arbitrary subset of the body of a rule, with all variables existentially quantified and the constants, if any, left unchanged.

Using this family of queries, we obtain the following characterization of uniform first-order rewritability for frontier-guarded TGDs.

Proposition 6.3. *Let Σ be a set of frontier-guarded TGDs and let S be a data schema. The following statements are equivalent:*

1. *Σ is uniformly first-order rewritable relative to S;*
2. *for every query $Q \in \mathcal{B}(\Sigma)$, the ontology-mediated query (S, Σ, Q) admits a first-order rewriting.*

This result shows that, although the characterization used for guarded TGDs cannot be transferred directly to the frontier-guarded setting, uniform rewritability can still be reduced to the rewritability of a specific finite family of queries derived from the bodies of the rules.

We will not develop the proof of Proposition 6.3 in this report. This characterization is part of ongoing work and is intended to be presented in a subsequent article, potentially together with a more general result.

For the remainder of this report, we instead focus on the second problem introduced earlier: the uniform termination of the semi-oblivious chase. To study this problem for frontier-guarded TGDs, we develop a new linearization procedure adapted to this larger class of rules. We will show that this transformation preserves the relevant termination behavior of the chase and use it to establish the complexity of uniform chase termination for frontier-guarded TGDs.

7 Linearization of Frontier-Guarded TGDs

We now introduce the linearization procedure that will be used to study chase termination for frontier-guarded TGDs. As in the guarded case, the objective is to replace a set of rules by a set of linear TGDs while preserving the behavior of the original chase. The construction is, however, more involved. Indeed, in a frontier-guarded rule, the guard is only required to contain the frontier variables, and therefore the applicability of a rule cannot in general be determined from the guard atom alone.

To overcome this difficulty, we associate with every atom a richer notion of type, recording not only its predicate and equality pattern, but also the configurations of rule bodies that can be realized around it in the chase. For simplicity, we assume throughout this section that TGDs have atomic heads. The complete formal construction and its technical details are given in Appendix A.

Σ-types. Let D be a database, let Σ be a set of frontier-guarded TGDs, and let

$$\alpha = R(t_1, \dots, t_k)$$

be an atom of $\mathrm{chase}(D, \Sigma)$. Its Σ-*type* with respect to D, denoted $t_D^\Sigma(\alpha)$, is a tuple

$$t_D^\Sigma(\alpha) = (R, \sim, \mathcal{G}),$$

where R is the predicate of α, $\sim$ records the equality pattern of its arguments, that is,

$$i \sim j \quad \Longleftrightarrow \quad t_i = t_j,$$

and $\mathcal{G}$ records the rule-body configurations that can be realized around α.

More precisely, $\mathcal{G}$ contains a pair (G, π) whenever there exists a rule $\sigma \in \Sigma$, a subset $G \subseteq \mathrm{body}(\sigma)$, and a homomorphism

$$h : G \to \mathrm{chase}(D, \Sigma)$$

such that π associates some terms of G with positions of α and

$$h(x) = \alpha[\pi(x)]$$

for every x on which π is defined.

Thus, the component $\mathcal{G}$ records which fragments of rule bodies can occur in the chase while fixing some of their terms to arguments of α. This additional information is precisely what is needed in the frontier-guarded setting, where atoms outside the frontier guard may be required to witness the applicability of a rule.

There are at most doubly exponentially many different Σ-types. For every type $t = (R, \sim, \mathcal{G})$, we introduce a fresh predicate p_t. Its arity is the number of equivalence classes of $\sim$.

Linearized database. For an atom

$$\alpha = R(t_1, \dots, t_k) \in D,$$

let $(t'_1, \dots, t'_j)$ be obtained by keeping the first occurrence of each distinct term of α, in their order of appearance. We encode α by the type atom

$$p_{t_D^\Sigma(\alpha)}(t'_1, \dots, t'_j).$$

The *linearized database*, denoted $\mathrm{lin}(D)$, is the set of all such type atoms.

The linearized rule set is composed of three families of rules.

Origin reconstruction rules. The first family simply recovers the original atom represented by a type atom. If $t = (R, \sim, \mathcal{G})$ and R has arity k, we introduce a rule

$$p_t(x_1, \dots, x_j) \to R(x_{\mathrm{rank}(1)}, \dots, x_{\mathrm{rank}(k)}),$$

where $\mathrm{rank}(i)$ identifies the equivalence class of position i according to its first occurrence. These rules reconstruct exactly the original database from its linearization.

Saturation reconstruction rules. The second family reconstructs consequences that can be obtained without introducing new terms. Suppose that σ is a Datalog rule and that a type $t = (R, \sim, \mathcal{G})$ contains $(\mathrm{body}(\sigma), \pi)$, where π is defined on the frontier variables of σ. The information stored in the type then guarantees that the body of σ can be realized while fixing its frontier to the corresponding positions of the represented atom. We can therefore introduce a linear rule

$$p_t(\bar{x}) \to P(\bar{x}_\pi),$$

where $P(\bar{x}_\pi)$ is the corresponding instantiation of the head of σ.

Together with the origin reconstruction rules, these rules recover the part of the original chase whose terms already occur in the database.

Complex rules. It remains to simulate rule applications that introduce fresh nulls. Suppose that an application of a rule $\sigma \in \Sigma$, whose frontier is guarded by an atom α_1 of type t_1, creates an atom α_2 of type t_2. We introduce a linear TGD of the form

$$p_{t_1}(\bar{x}) \to \exists \bar{z}\, p_{t_2}(\bar{y}).$$

The tuple $\bar{y}$ preserves the frontier terms inherited from α_1, while positions corresponding to existential variables of σ are filled with fresh existential variables from $\bar{z}$.

The crucial point is that this construction depends only on the types t_1 and t_2, and not on the particular atoms α_1 and α_2 used to define it. More precisely, if an application from an atom of type t_1 produces an atom of type t_2, then the same transition can be reproduced from any atom of type t_1, producing an atom of type t_2. The proof of this property, which is the main technical part of the construction, is deferred to Appendix A.

We denote respectively by Σ_{or}, Σ_{sr} and Σ_c the sets of origin reconstruction, saturation reconstruction and complex TGDs, and define

$$\mathrm{lin}(\Sigma) = \Sigma_{or} \cup \Sigma_{sr} \cup \Sigma_c.$$

By construction, every rule of $\mathrm{lin}(\Sigma)$ has a single type atom in its body and is therefore linear.

The correctness of the construction is summarized by the following proposition.

Proposition 7.1. *Let Σ be a set of frontier-guarded TGDs and let D be a database over a schema $\mathcal{S}$. Then*

$$\mathrm{chase}(D, \Sigma) \quad \textit{and} \quad \mathrm{chase}(\mathrm{lin}(D), \mathrm{lin}(\Sigma))\big|_{\mathcal{S}}$$

are homomorphically equivalent.

The full definition of the linearization, together with the proof establishing Proposition 7.1, is presented in Appendix A.

8 Uniform Termination of the Semi-Oblivious Chase

We now turn to the proof of Theorem 3.4. The lower bound is already known from the literature [2], where it is shown that $\mathsf{UniformTermination(G)}$ is 2-ExpTime-hard. Therefore, it remains to establish the matching upper bound for frontier-guarded TGDs.

Our proof relies on the linearization procedure introduced in the previous section. The main idea is that, instead of considering the semi-oblivious chase on every possible database, it is sufficient to study the chase of a single, canonical database after linearization. This considerably simplifies the problem while preserving the termination behavior of the chase.

To formalize this idea, we use the notion of a *critical database*. Given a schema S, the critical database of S, denoted D_S, contains one fact for every relation symbol of the schema, where every argument is replaced by the same arbitrary constant c. Formally,

$$D_S = \{\, R(\underbrace{c, \ldots, c}_{\mathrm{ar}(R)}) \mid R \in S \,\}.$$

The key characterization underlying our approach is the following.

Proposition 8.1. *Let Σ be a set of frontier-guarded TGDs and let S be a data schema. The following statements are equivalent:*

1. *Σ ensures the uniform termination of the semi-oblivious chase relative to S;*
2. *the chase* $\mathrm{chase}(\mathrm{Lin}(D_S), \mathrm{Lin}(\Sigma))$ *is finite.*

Before proving this characterization, let us explain how it yields the upper bound of Theorem 3.4. By Proposition 8.1, deciding whether Σ is uniformly terminating reduces to constructing the linearized database $\mathrm{Lin}(D_S)$ and the linearized set of TGDs $\mathrm{Lin}(\Sigma)$, and then checking whether the chase of the former with respect to the latter is finite.

The critical database can clearly be constructed in polynomial time. Moreover, Section A shows that both $\mathrm{Lin}(D_S)$ and $\mathrm{Lin}(\Sigma)$ can be computed in double-exponential time. Since $\mathrm{Lin}(\Sigma)$ is a set of simple-linear TGDs, the finiteness of $\mathrm{chase}(\mathrm{Lin}(D_S), \mathrm{Lin}(\Sigma))$ can be decided in polynomial time in the size of the linearized instance and rule set [8]. Consequently, the overall procedure runs in double-exponential time, proving the desired upper bound.

8.1 Proof of Proposition 8.1

The proof of Proposition 8.1 follows from two independent ingredients.

The first is a classical result stating that, to decide whether a set of TGDs guarantees uniform termination over a schema, it is sufficient to consider the critical database of that schema.

Lemma 8.2. *Let S be a schema and Σ a set of TGDs. The following are equivalent:*

1. *Σ ensures the uniform termination of the semi-oblivious chase relative to S;*
2. $\mathrm{chase}(D_S, \Sigma)$ *is finite.*

The second ingredient shows that the linearization procedure preserves the termination of the semi-oblivious chase.

Lemma 8.3. *Let D be a database and Σ a set of frontier-guarded TGDs. The following are equivalent:*

1. $\mathrm{chase}(D, \Sigma)$ *is finite;*
2. $\mathrm{chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma))$ *is finite.*

Proof. We prove both implications. Recall that

$$\mathrm{Lin}(\Sigma) = \Sigma_{or} \cup \Sigma_{sr} \cup \Sigma_c,$$

where Σ_{or} contains the origin reconstruction rules, Σ_{sr} the saturation reconstruction rules, and Σ_c the complex TGDs.

The important observation is that rules in $\Sigma_{or} \cup \Sigma_{sr}$ do not introduce fresh nulls. Hence, from a finite set of terms, they can derive only finitely many atoms. As a consequence, an infinite chase under $\mathrm{Lin}(\Sigma)$ must contain infinitely many applications of complex TGDs.

We also use the following property of the construction, established in Section 7: if an application of a rule $\sigma \in \Sigma$ whose frontier is guarded by an atom of type t_1 creates an atom of type t_2, then the corresponding complex TGD

$$p_{t_1}(\bar{x}) \to \exists \bar{z}\, p_{t_2}(\bar{y})$$

belongs to Σ_c. Conversely, whenever this complex TGD is applied to a type atom representing an atom α_1 of type t_1, the application can be realized in the original chase by an application of σ whose frontier is mapped to α_1, producing an atom of type t_2.

($\Rightarrow$) Assume that $\mathrm{chase}(D, \Sigma)$ is infinite.

Since Σ is finite and predicates have finite arity, an infinite chase cannot contain only finitely many terms: over a finite set of terms, only finitely many atoms can be formed. Therefore, $\mathrm{chase}(D, \Sigma)$ contains infinitely many applications of existential TGDs introducing fresh nulls.

We show inductively that every atom created by such an application has a corresponding type atom in

$$\mathrm{chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma)).$$

For atoms of D, this holds by definition of $\mathrm{Lin}(D)$. Namely, for every $\alpha \in D$, the atom

$$p_{t^{\Sigma}_{D}(\alpha)}(\bar{a})$$

belongs to $\mathrm{Lin}(D)$, where $\bar{a}$ consists of the distinct arguments of α in order of first occurrence.

Assume now that an atom α_2 is created in the original chase by an application of a TGD σ, and let α_1 be an atom guarding the image of the frontier of σ. Suppose that α_1 has type t_1 and that α_2 has type t_2.

By induction, the type atom corresponding to α_1 occurs in the linearized chase. By construction of Σ_c, there is a complex TGD

$$p_{t_1}(\bar{x}) \to \exists \bar{z}\, p_{t_2}(\bar{y})$$

simulating this application. Hence the corresponding type atom for α_2 is generated in the linearized chase.

Thus every existential creation step of the original chase is simulated by a complex-rule application in the linearized chase. Since there are infinitely many such creation steps, the linearized chase contains infinitely many applications of complex TGDs and is therefore infinite. Hence

$$\mathrm{chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma)) \text{ is infinite.}$$

($\Leftarrow$) Assume now that

$$\mathrm{chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma))$$

is infinite.

As observed above, the rules in $\Sigma_{or} \cup \Sigma_{sr}$ introduce no fresh nulls. If only finitely many complex TGDs were applied, then only finitely many terms would occur in the linearized chase. Since the rule set and the schema are finite, only finitely many atoms could then be generated. This would contradict the assumption that the chase is infinite.

Consequently, infinitely many complex TGDs are applied.

We now show inductively that every type atom generated through complex TGDs can be realized by an atom of the original chase having the same Σ-type.

The type atoms initially occurring in $\mathrm{Lin}(D)$ directly correspond to atoms of D, so the property holds at the beginning of the chase.

Suppose that

$$p_{t_2}(\bar{b})$$

is generated from

$$p_{t_1}(\bar{a})$$

by a complex TGD

$$p_{t_1}(\bar{x}) \to \exists \bar{z}\, p_{t_2}(\bar{y}),$$

and assume inductively that $p_{t_1}(\bar{a})$ is represented by an atom $\alpha_1 \in \mathrm{chase}(D, \Sigma)$ of type t_1.

By the type-transition property of the linearization, the complex TGD comes from an original TGD $\sigma \in \Sigma$ that can be applied with its frontier guarded by α_1. Moreover, this application produces an atom α_2 whose Σ-type is exactly t_2. We therefore associate $p_{t_2}(\bar{b})$ with this newly produced atom α_2.

Fresh nulls introduced by the complex TGD are associated with the fresh nulls introduced by the corresponding application of σ. Thus the construction can be continued along the whole sequence of complex rule applications.

Since the semi-oblivious chase identifies triggers according to their restriction to frontier variables, distinct active applications of a complex TGD correspond to distinct frontier mappings. The construction of complex TGDs preserves precisely these frontier arguments. Consequently, infinitely many active complex-rule applications yield infinitely many distinct existential applications in the original semi-oblivious chase.

It follows that $\mathrm{chase}(D, \Sigma)$ contains infinitely many fresh nulls, and hence infinitely many atoms. Therefore,

$$\mathrm{chase}(D, \Sigma) \text{ is infinite.}$$

We have proved

$$\mathrm{chase}(D, \Sigma) \text{ is finite} \quad \Longleftrightarrow \quad \mathrm{chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma)) \text{ is finite.}$$

$\square$

Combining Lemmas 8.2 and 8.3 immediately yields Proposition 8.1. Indeed, by Lemma 8.2, Σ ensures the uniform termination of the semi-oblivious chase relative to S if and only if

$$\mathrm{chase}(D_S, \Sigma)$$

is finite. Applying Lemma 8.3 to the critical database D_S, this is equivalent to

$$\mathrm{chase}(\mathrm{Lin}(D_S), \mathrm{Lin}(\Sigma))$$

being finite. Therefore,

$$\Sigma \text{ uniformly terminates relative to } S \quad \Longleftrightarrow \quad \mathrm{chase}(\mathrm{Lin}(D_S), \mathrm{Lin}(\Sigma)) \text{ is finite,}$$

which is precisely Proposition 8.1.

9 Conclusion and Future Work

In this report, we studied two static-analysis problems for existential rules: uniform first-order rewritability and uniform termination of the semi-oblivious chase. Our objective was to understand these problems first for guarded TGDs and then for the more general class of frontier-guarded TGDs.

For uniform first-order rewritability, we first considered guarded TGDs. Using a linearization into linear TGDs, we showed that conjunctive-query answering over complete databases can be reduced to the evaluation of a union of conjunctive queries. This allowed us to characterize uniform first-order rewritability in terms of atomic queries: a set of guarded TGDs is uniformly first-order rewritable if and only if all atomic queries are first-order rewritable. Together with existing results for atomic-query rewritability and the lower-bound reduction presented in Section 5.2, this yields the desired complexity result.

We then investigated the extension of this approach to frontier-guarded TGDs. The main intermediate property used in the guarded case fails in this more general setting: conjunctive queries are no longer necessarily UCQ-rewritable over complete databases. Nevertheless, this does not prevent a characterization of uniform first-order rewritability. As stated in Proposition 6.3, atomic queries can be replaced by the finite family of sub-body queries $B(\Sigma)$. The proof of this result was not developed in this report and constitutes part of our ongoing work.

Our second direction concerned uniform termination of the semi-oblivious chase. To study this problem for frontier-guarded TGDs, we introduced a new linearization procedure based on richer types that record local configurations of rule bodies. We showed that this transformation preserves chase termination. Combined with the critical-database property and known termination results for simple-linear TGDs, this gives a 2-ExpTime upper bound for uniform termination of frontier-guarded TGDs, matching the known lower bound for guarded TGDs.

Several questions arising from this work remain to be investigated. In particular, we intend to further develop the frontier-guarded rewritability results presented in Section 6. The characterization through sub-body queries suggests that the techniques developed here may extend beyond the precise setting considered in this report. We hope to strengthen these results, potentially obtaining a more general characterization, and to present the complete proofs together with the linearization techniques in a future research article.

More broadly, the results of this internship show that moving from guarded to frontier-guarded TGDs requires keeping track of substantially more information about the interaction between rule bodies. Understanding how this additional structure can be represented while retaining decidable and computationally manageable static-analysis problems remains a natural direction for future work.

Acknowledgements

I would like to thank Andreas Pieris, Aldo Ricioppo, and Michaël Thomazo for their help, guidance, and the many valuable discussions throughout this internship.

References

[1] Stanislav Kikot, Roman Kontchakov, Vladimir V. Podolskii, and Michael Zakharyaschev. On the price of query rewriting in ontology-based data access. *Artificial Intelligence*, 213:42–59, 2014.

[2] Marco Calautti, Georg Gottlob, and Andreas Pieris. Chase termination for guarded existential rules. In *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*, pages 91–103, 2015.

[3] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Artificial Intelligence Research*, 48:115–174, 2013.

[4] Georg Gottlob, Marco Manna, and Andreas Pieris. Polynomial combined rewritings for existential rules. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014*, pages 268–277. AAAI Press, 2014.

[5] Marco Calautti, Georg Gottlob, and Andreas Pieris. Chase termination for guarded existential rules. In *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS 2015)*, pages 91–103. ACM, 2015.

[6] Pablo Barceló, Gerald Berger, Carsten Lutz, and Andreas Pieris. First-order rewritability of frontier-guarded ontology-mediated queries. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 1707–1713, 2018.

[7] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Artificial Intelligence Research*, 48:115–174, 2013.

[8] Marco Calautti, Georg Gottlob, and Andreas Pieris. Non-uniformly terminating chase: Size and complexity. In *PODS*, pages 369–378, 2022.

A Linearization of Frontier-Guarded Tuple-Generationg Dependencies

The linearization construction presented in this section is due to Michael Thomazo. The proof of the main correctness proposition stated at the end of the section, however, was developed as part of this internship with his help and the help of my supervisor.

A.1 Presentation of the Linerization

To simplify the presentation, we assume TGDs to be with atomic head. This is done without loss of generality .

Definition A.1 (Σ-type)**.** *Let D be a database, Σ be a set of frontier-guarded TGDs. Let α be an atom of* $\text{chase}\, D\Sigma$. *The* Σ-type *of α with respect to D and Σ, denoted by $t_D^\Sigma(\alpha)$, is the tuple $(R, \sim, \mathcal{G})$ where:*

- *R is the predicate symbol of α;*
- *$\sim$ is the equivalence relation on the positions of α such that for all i, j, $i \sim j$ if and only if $\alpha[i] = \alpha[j]$*
- *$\mathcal{G}$ is the set of all pairs (G, π) for which there exists a TGD $\sigma \in \Sigma$ such that:*
  - *$G \subseteq \text{body}(\sigma)$;*
  - *π is a mapping from a subset $S \subseteq \mathsf{terms}(G)$ to the positions of α;*
  - *there exists a homomorphism $h : G \to \text{chase}\, D\Sigma$ such that $h(t) = \alpha[\pi(t)]$ for all $t \in S$.*

Note that there is a double-exponential number of Σ-types. For each Σ-type $t = (R, \sim, \mathcal{G})$, we introduce a fresh predicate p_t of arity the number of $\sim$-classes. The following proposition is a corollary of CQ entailment being solvable in double-exponential time under frontier-guarded TGDs.

Proposition A.2. *For any atom α in D, the $t_D^\Sigma(\alpha)$ is computable in double-exponential time.*

Linearization simply collects all the Σ-types for database atoms.

Definition A.3 (Linearized Database)**.** *Let D be a database and Σ be a set of frontier-guarded TGDs. The* linearized database *of D w.r.t. Σ is defined as follows: for each atom $\alpha(\mathbf{x}) \in D$, let $\mathbf{t_x}$ be the tuple formed by the first occurence of each distinct term in $\mathbf{x}$, preserving their original order. Then the linearization of D is $\{p_{t_D^\Sigma(\alpha)}(\mathbf{t_x}) \mid \alpha(\mathbf{x}) \in D\}$.*

The following notion of rank is merely a technical tool to describe our construction.

Definition A.4 (Rank)**.** *Let $\sim$ be an equivalence over $\{1, \ldots, k\}$ for some $k \in \mathbb{N}$. For each equivalence class C under $\sim$, define $\mathsf{min}(C) = \mathsf{min}\{i \mid i \in C\}$. The* rank *of C is $\mathsf{rank}_\sim(C) = 1 + |\{C' \in \{1, \ldots, k\}_{/\sim}$ such that $\mathsf{min}(C') < \mathsf{min}(C)\}|$. We define the rank of an element of $\{1, \ldots, k\}$ as the rank of its equivalence class.*

Definition A.5 (Origin Reconstruction TGD)**.** *Let $(R, \sim, \mathcal{G})$ be a dependency type, where R is a predicate of arity k. For each $i \in \{1, \ldots, k\}$, let $\mathsf{rank}(i)$ be the rank of its dependency class under $\sim$. The* origin reconstruction TGD *associated with $(R, \sim)$ is $\forall x_1 \ldots x_j\ p_{R,\sim,\mathcal{G}}(x_1, \ldots, x_j) \to R(x_{\mathsf{rank}(1)}, \ldots, x_{\mathsf{rank}(k)})$, where j is the number of equivalence class under $\sim$.*

The following proposition illustrates the interplay between origin reconstruction TGDs and linearization.

Proposition A.6. *Let Σ be a set of frontier-guarded TGDs, D be a database with signature $\mathcal{S}$, and D' be the linearization of D w.r.t. Σ. Let Σ_{or} be the set of all origin reconstruction TGDs associated with Σ-types. Then* $\text{chase}\, D'\Sigma_{or|\mathcal{S}} = D$.

Proof. For every atom $\alpha = R(t_1, \ldots, t_k) \in D$, let $t_D^\Sigma(\alpha) = (R, \sim, \mathcal{G})$ and let $(t'_1, \ldots, t'_j)$ be the tuple consisting of the first occurrence of each distinct term of α, in their order of appearance. By definition of the linearized database, $p_{R,\sim,\mathcal{G}}(t'_1, \ldots, t'_j) \in D'$.

Let $p_{R,\sim,\mathcal{G}}(x_1, \ldots, x_j) \to R(x_{\mathsf{rank}(1)}, \ldots, x_{\mathsf{rank}(k)})$ be the origin reconstruction TGD associated with the dependency type of α.

For every position $i \in \{1, \ldots, k\}$, the rank of i is precisely the position of the equivalence class of i under $\sim$ when the classes are ordered by the first occurrence of one of their elements. Since $(t'_1, \ldots, t'_j)$ was obtained by keeping exactly the first occurrence of every distinct term of α, it follows that $t'_{\mathsf{rank}(i)} = \alpha[i]$ for every position i. Consequently, applying the origin reconstruction TGD to $p_{R,\sim,\mathcal{G}}(t'_1, \ldots, t'_j)$ produces exactly the atom $R(\alpha[1], \ldots, \alpha[k]) = \alpha$. Since this argument applies to every atom of D, we obtain $D \subseteq \text{chase}\, D'\Sigma_{or|\mathcal{S}}$.

Conversely, every TGD of Σ_{or} has a head over the original signature and a body consisting of a single type predicate. By construction of the linearized database, every type atom occurring in D' corresponds to an atom of D, and the associated origin reconstruction TGD reconstructs precisely that atom. Hence no atom over the original signature other than those of D can be produced.

Therefore, $\text{chase}\, D'\Sigma_{or|\mathcal{S}} = D$. ☐

Definition A.7 (Saturation Reconstruction TGD)**.** *Let σ be a frontier-guarded Datalog TGD with head $p(y_1, \ldots, y_k)$. Let $(R, \sim, \mathcal{G})$ be a dependency type such that $\mathcal{G}$ contains* $(\text{body}(\sigma), \pi)$ *for some mapping π whose domain includes* $\text{fr}(\sigma)$ *(hence, y_1 to y_k). The* saturation reconstruction TGD *associated with this Σ-type, σ and π is $\forall x_1, \ldots, x_j\ p_{R,\sim,\mathcal{G}}(x_1, \ldots, x_j) \to p(x_{\pi(y_1)}, \ldots, x_{\pi(y_k)})$, where j is the number of equivlance classes of $\sim$.*

The following proposition illustrates the interplay between the whole set of reconstruction TGDs and linearization.

Proposition A.8. *Let Σ be a set of frontier-guarded TGDs, D be a database with signature $\mathcal{S}$, and D' be the linearization of D w.r.t. Σ. Let Σ_{or} be the set of all origin reconstruction TGDs associated with Σ-types, and Σ_{sr} be the set of all saturation reconstruction TGDs associated with Σ. Then* $\text{chase}\, D'\Sigma_{or} \cup \Sigma_{sr|\mathcal{S}} = \text{chase}\, D\Sigma_{\mathsf{terms}(D)}$.

Proof. By the previous proposition, $\text{chase}\, D'\Sigma_{or|\mathcal{S}} = D$. As saturation reconstruction TGDs and orginal reconstruction TGDs do not interact, it remains to show that the saturation reconstruction TGDs reconstruct exactly the atoms of $\text{chase}\, D\Sigma$ whose arguments belong to $\mathsf{terms}(D)$.

We first prove $\text{chase}\, D\Sigma_{|\mathsf{terms}(D)} \subseteq \text{chase}\, D'\Sigma_{or} \cup \Sigma_{sr|\mathcal{S}}$. Let $p(t_1, \ldots, t_k) \in \text{chase}\, D\Sigma_{|\mathsf{terms}(D)}$. Then there exists a frontier-guarded TGD $\sigma : \text{body}(\sigma) \to p(y_1, \ldots, y_k)$ and a homomorphism $h : \text{body}(\sigma) \to \text{chase}\, D\Sigma$ such that $h(y_i) = t_i$ for every i. Since σ is frontier-guarded, the frontier variables all occur in the guard atom of the body. Let α be a database atom guarding the image the frontier, and let $(R, \sim, \mathcal{G}) = t_D^\Sigma(\alpha)$. By definition of dependency type,$(\text{body}(\sigma), \pi) \in \mathcal{G}$, where π maps every frontier variable y_i to the position of $h(y_i)$ in α. Since α is a database atom, $p_{t_D^\Sigma(\alpha)}(t_{\mathbf{x}})$ belongs to D', where $t_{\mathbf{x}}$ is the tuble formed by the first occurence of each distinct term in α, preserving their original order. Therefore the corresponding saturation reconstruction TGD is applicable. It derives $p(t_1, \ldots, t_k)$ because the variables occurring in the body of the reconstruction TGD are instantiated with the arguments of α, and π reproduces exactly the values of the frontier variables under h.

We now prove the converse inclusion.

Every atom over the original signature derived by a saturation reconstruction TGD is obtained from a type predicate $p_{R,\sim,\mathcal{G}}(a_1, \ldots, a_j)$ already present in D', and from a TGD associated with a pair $(\mathrm{body}(\sigma), \pi) \in \mathcal{G}$. By definition of $\mathcal{G}$, there exists a homomorphism $h : \mathrm{body}(\sigma) \to \mathrm{chase}\, D\Sigma$, mapping every variable in the domain of π to the corresponding position of the guard atom. Since the head of a Datalog TGD contains only frontier variables, the atom produced by the saturation reconstruction TGD is exactly the head of σ under h. Therefore it already belongs to $\mathrm{chase}\, D\Sigma$.

Moreover, saturation reconstruction TGDs introduce no existential variables, so every derived atom contains only terms of D. Hence $\mathrm{chase}\, D'\Sigma_{or} \cup \Sigma_{sr|\mathcal{S}} \subseteq \mathrm{chase}\, D\Sigma_{\mathsf{terms}(D)}$.

Combining the two inclusions yields the result.

□

Definition A.9 (Complex TGD). *Let α_1 of type $t_1 = (R_1, \sim_1, \mathcal{G}_1)$ and α_2 of type $t_2 = (R_2, \sim_2, \mathcal{G}_2)$ be two atoms of* $\mathrm{chase}\, D\Sigma$ *such that there exists a TGD σ with head atom α_h, and a homomorphism π from* $\mathrm{body}(\sigma)$ *to* $\mathrm{chase}\, D\Sigma$ *such that:*

- *α_1 guards $\pi(\mathrm{fr}(\sigma))$;*
- *α_2 is the atom resulting from this TGD application.*

The complex TGD *associated with α_1 and α_2 is the TGD:*

$$p_{t_1}(x_1, \ldots, x_j) \to \exists \mathbf{z}\; p_{t_2}(y_1, \ldots, y_l)$$

where $\mathbf{z}$ is a tuple of fresh existential variables, and for each $i \in \{1, \ldots, l\}$:

- *if there exists m such that $\alpha_h[m]$ is an existentially quantified variable and $\mathsf{rank}_{\sim_2}(m) = i$, then y_i is an existentially quantified variable from $\mathbf{z}$;*
- *otherwise, if there are two positions i' and m' such that $\mathsf{rank}_{\sim_2}(i') = i$, and $\mathsf{rank}_{\sim_1}(m') = m$ and $\pi(\alpha_h[i']) = \alpha_1[m']$, then $y_i = x_m$.*

To understand the intuition behind the above complex TGDs, let us consider the following proposition.

Proposition A.10. *Let $\alpha_1 \in \mathrm{chase}\, D\Sigma$ be an atom of Σ-type $t_1 = (R_1, \sim_1, \mathcal{G}_1)$, and $\alpha_2 \in \mathrm{chase}\, D\Sigma$ of Σ-type $t_2 = (R_2, \sim_2, \mathcal{G}_2)$ obtained by applying the TGD $\sigma \in \Sigma$ by π where the guard of σ is mapped to α_1. Let α_1' be an arbitrary atom of Σ-type t_1, and let ψ be the mapping that associated to the i^{th} argument of α_1 the i^{th} argument of α_1'. Then σ is applicable by an extension of $(\psi \circ \pi)_{\mathrm{fr}(\sigma)}$ that maps the guard of σ to α_1', and the atom α_2' thereby created is of Σ-type t_2.*

Proof. **ψ is well defined.**

Since α_1 and α_1' have the same Σ-type $t_1 = (R_1, \sim_1, \mathcal{G}_1)$, they have the same predicate R_1 and the same arity. Moreover, $\sim_1$ encodes the equality pattern among their arguments, and thus $\alpha_1[i] = \alpha_1[j]$ if and only if $\alpha_1'[i] = \alpha_1'[j]$. Hence ψ is well-defined.

σ is applicable via an extension of $\psi \circ \pi$. Since α_2 is obtained by applying σ to α_1, the pair $(\mathrm{body}(\sigma), \pi_p)$ belongs to $\mathcal{G}_1$, where π_p is such that for any $t \in \mathrm{fr}(\sigma)$, $\alpha_2[\pi_p(t)] = \pi(t)$. As α_1' has the same type as α_1, there exists a homomorphism

$$h' : \mathrm{body}(\sigma) \to \mathrm{chase}\, D\Sigma$$

that agrees with $\psi \circ \pi$ on every frontier variable of σ. Thus h' is a trigger of σ whose restriction to the frontier is precisely $\psi \circ \pi$. Applying σ with h' produces an atom α_2'.

α_2' **has the same predicate, the same equality pattern as** α_2**.** Both atoms are produced by the same TGD σ, so they have the same predicate. Moreover, frontier variables are identified according to ψ, which preserves the equality pattern of α_1, while existential variables are fresh in both applications. Hence two positions of the head of σ are equal in α_2 if and only if they are equal in α_2'. Therefore α_2 and α_2' have the same first two components $(R_2, \sim_2)$ of their dependency types.

The third components of Σ-types of α_2 and α_2' are equal. We show that $\mathcal{G}_2 \subseteq \mathcal{G}_2'$, the converse follows by exchanging the roles of α_1, α_2 and α_1' and α_2'. Let $(G, \pi_2) \in \mathcal{G}(\alpha_2)$, witnessed by a homomorphism $h : G \to \text{chase}\, D\Sigma$ such that $h(t) = \alpha_2[\pi_2(t)]$ whenever $\pi_2(t)$ is defined. Partition the atoms of G into two sets:

$$G_\downarrow = \{\beta \in G \mid h(\beta) \text{ contains a descendant of a term generated in } \alpha_2\},$$

and $G_\uparrow = G \setminus G_\downarrow$. Since Σ is frontier-guarded, every path from an atom of $G_\uparrow$ to an atom of $G_\downarrow$ passes through a frontier term of α_2. Hence every term shared by $G_\uparrow$ and $G_\downarrow$ is mapped by h to a frontier term of α_2. We can thus assume without loss of generality that π_2 is defined on all such variables. Let $\pi_\uparrow$ be the restriction of π_2 to the terms of $G_\uparrow$ that are mapped to frontier positions of α_2. Then $(G_\uparrow, \pi_\uparrow) \in \mathcal{G}_2(\alpha_2)$, and since the frontier terms of α_2 also belong to α_1, $(G_\uparrow, \pi_\uparrow') \in \mathcal{G}_1$, where $\pi_\uparrow'(t)$ is defined when $\pi_\uparrow(t)$ is, and is such that $\alpha_2[\pi_\uparrow(t)] = \alpha_1[\pi_\uparrow'(t)]$. Hence, as α_1 and α_1' have the same type, there exists h' that maps $G_\uparrow$ into $\text{chase}\, D\Sigma$ mapping the relevant terms (including those shared with $G_\downarrow$) into the frontier terms of α_2' according to $\pi_\uparrow'$.

It remains to show that h' can be extented to $G_\downarrow$. Consider an arbitrary chase sequence, and enumerate the descendants of α_2 as $\gamma_0 = \alpha_2, \gamma_1, \ldots$ according to their order of their first creation in the chase sequence (an atom may have been generated several times due to full TGDs). We show that there exists in $\text{chase}\, D\Sigma$ $\gamma_0', \gamma_1', \ldots$ descendants of α_2' and a homomorphism φ_i from $\Gamma_i = \{\gamma_0, \ldots, \gamma_i\}$ onto $\Gamma_i' = \{\gamma_0', \ldots, \gamma_i'\}$, such that for each $i > 0$, φ_i extends φ_{i-1}, and every strict descendant of the frontier of α_2 are mapped to strict descendant of the frontier of α_2'.

Base case: $\varphi_0 = \psi$

Inductive case: let us assume φ_i is defined, and let us show that γ_{i+1}' can be generated. γ_{i+1} is created by a rule application σ_{i+1}, π_{i+1}. Let $\text{body}(\sigma_{i+1})$ be partitioned into $B_\downarrow = \pi_{i+1}^{-1}(\Gamma_i)$ and $B_\uparrow = \text{body}(\sigma_{i+1}) \setminus B_\downarrow$. As Σ is frontier-guarded, it holds that all terms shared by $B_\uparrow$ and $B_\downarrow$ are mapped by π_{i+1} to $\mathsf{terms}(\alpha_2) \cap \mathsf{terms}(\alpha_1)$. Hence $(B_\uparrow, \pi_\uparrow)$, where $\pi_\uparrow$ maps the shared terms to the corresponding positions of α_1, belongs to the type of atom_1, and thus to the type of atom_1'. Thus, there exists a homomorphism $h_\uparrow$ from $B_\uparrow$ to $\text{chase}\, D\Sigma$ agreeing with $\psi \circ \pi_{i+1}$ on all terms shared with $B_\downarrow$.

Define $h_\downarrow$ by $h_\downarrow = (\varphi_i \circ \pi_{i+1})_{|\mathsf{terms}(B_\downarrow)}$. By construction of φ_i, this is a valid homomorphism from $B_\downarrow$ to Γ_i'. $h_\uparrow$ and $h_\downarrow$ are compatible, as the terms shared by $B_\uparrow$ and $B_\downarrow$ are mapped through ψ to the frontier terms of α_2'. Hence $h_\downarrow \cup h_\uparrow$ is a valid homomorphism from $\text{body}(\sigma_{i+1})$ to $\text{chase}\, D\Sigma$, forming a trigger for σ_{i+1}. Let γ_{i+1}' be the atom generated by it.

We extend φ_i to φ_{i+1} by setting $\varphi_{i+1}(\gamma_{i+1}[j]) = \gamma_{i+1}'[j]$ for each position j.

By considering Γ_k such that $G_\downarrow$ maps to Γ_k by h, it holds that $G_\downarrow$ maps to Γ_k' by $\varphi_k \circ h$. By construction $\varphi_k \circ h$ and h' agree on terms shared between $G_\uparrow$ and $G_\downarrow$, which shows that $\mathcal{G}_2 \subseteq \mathcal{G}_2'$, and concludes the proof. □

A.2 Proof of the corectness

Proposition A.11. *Let Σ be a set of frontier-guarded TGDs, D a database with signature $\mathcal{S}$. Let $\text{Lin}(\Sigma) = \Sigma_{or} \cup \Sigma_{sr} \cup \Sigma_c$. Then $\text{chase}\, D\Sigma$ is homomorphically equivalent to $\text{chase}\, \text{Lin}(D)\text{Lin}(\Sigma)_{|\mathcal{S}}$.*

Proof. Let

$$I = \mathrm{Chase}(D, \Sigma) \qquad \text{and} \qquad J = \mathrm{Chase}(\mathrm{Lin}(D), \mathrm{Lin}(\Sigma)).$$

We prove the result by constructing homomorphisms

$$h : I \longrightarrow J_{|\mathcal{S}} \qquad \text{and} \qquad g : J_{|\mathcal{S}} \longrightarrow I.$$

For an atom

$$\alpha = R(a_1, \ldots, a_k) \in I,$$

let

$$t_D^\Sigma(\alpha) = (R, \sim, \mathcal{G}),$$

and denote by

$$\mathrm{red}(\alpha) = (a_{i_1}, \ldots, a_{i_j})$$

the tuple obtained by keeping the first occurrence of every distinct term of α, in their order of appearance. Thus, for every $r \in \{1, \ldots, k\}$,

$$a_r = \mathrm{red}(\alpha)[\mathsf{rank}_\sim(r)].$$

From the original chase to the linearized chase.

We construct h inductively along a chase derivation of I. At every stage, we maintain the following invariant: for every atom α already generated in the original chase, J contains the type atom

$$p_{t_D^\Sigma(\alpha)}\big(h(\mathrm{red}(\alpha))\big).$$

For the initial database, let $\alpha \in D$. By definition of $\mathrm{Lin}(D)$,

$$p_{t_D^\Sigma(\alpha)}\big(\mathrm{red}(\alpha)\big) \in \mathrm{Lin}(D).$$

Since homomorphisms are the identity on constants, we initially define $h(c) = c$ for every $c \in \mathsf{terms}(D)$. Hence the invariant holds for every atom of D.

Furthermore, the origin reconstruction TGD associated with $t_D^\Sigma(\alpha)$ reconstructs exactly α. Consequently,

$$D \subseteq J_{|\mathcal{S}}.$$

Assume now that the invariant holds after some number of chase steps, and consider a new atom α_2 obtained in the original chase by applying a TGD $\sigma \in \Sigma$ through a homomorphism

$$\pi : \mathrm{body}(\sigma) \longrightarrow I.$$

Since σ is frontier-guarded, the image of its frontier is contained in the image of a body atom. Let α_1 be such an atom, and write

$$t_1 = t_D^\Sigma(\alpha_1) \qquad \text{and} \qquad t_2 = t_D^\Sigma(\alpha_2).$$

We distinguish whether σ creates a new term.

First case: σ is a Datalog TGD.

In this case, no new null is introduced. By definition of the Σ-type of α_1, the component $\mathcal{G}_1$ records the existence of the homomorphism witnessing the application of σ. In particular,

$$(\mathrm{body}(\sigma), \pi_1) \in \mathcal{G}_1$$

for the corresponding positional mapping π_1.

By the induction hypothesis, the type atom corresponding to α_1 occurs in J. Hence the associated saturation reconstruction TGD can be applied. By construction of this TGD, the atom thereby produced is exactly

$$h(\alpha_2).$$

Thus

$$h(\alpha_2) \in J_{|\mathcal{S}}.$$

No extension of h is required, since no fresh term has been created.

Second case: σ introduces existentially quantified terms.

By definition of the complex TGDs, since an application of σ from an atom of type t_1 produces an atom of type t_2, the set Σ_c contains the complex TGD

$$p_{t_1}(x_1, \ldots, x_j) \to \exists \mathbf{z}\, p_{t_2}(y_1, \ldots, y_l)$$

associated with this transition.

By the induction hypothesis, J contains the type atom corresponding to α_1. We may therefore apply this complex TGD. Let

$$p_{t_2}(b_1, \ldots, b_l)$$

be the type atom obtained from this application.

By construction of the complex TGD, every argument of α_2 inherited from the frontier of σ is represented by the corresponding argument of the type atom of α_1. Consequently, its image is already determined by h.

For every null n freshly created in α_2, let n' be the corresponding fresh null introduced by the application of the complex TGD, and extend h by

$$h(n) = n'.$$

The equality relation $\sim_2$ stored in t_2 guarantees that this extension respects all equalities between positions of α_2. Hence

$$p_{t_2}\big(h(\mathrm{red}(\alpha_2))\big) \in J.$$

The invariant is therefore preserved.

Finally, applying the origin reconstruction TGD associated with t_2 gives

$$h(\alpha_2) \in J_{|\mathcal{S}}.$$

Thus, by induction, every atom of I is mapped to an atom of $J_{|\mathcal{S}}$. Since the mappings constructed at successive stages agree on all previously introduced terms, their union is a well-defined homomorphism

$$h : I \longrightarrow J_{|\mathcal{S}}.$$

From the linearized chase to the original chase.

We now construct a homomorphism

$$g : J_{|\mathcal{S}} \longrightarrow I.$$

For this direction, we proceed inductively along a chase derivation of J. We maintain the following invariant:

for every type atom

$$p_t(b_1, \ldots, b_j) \in J,$$

where $t = (R, \sim, \mathcal{G})$, there exists an atom

$$\alpha = R(a_1, \ldots, a_k) \in I$$

such that

$$t_D^{\Sigma}(\alpha) = t$$

and, writing

$$\mathrm{red}(\alpha) = (a_{i_1}, \ldots, a_{i_j}),$$

we have

$$g(b_r) = a_{i_r}$$

for every $r \in \{1, \ldots, j\}$.

We say in this case that α realizes the type atom $p_t(b_1, \ldots, b_j)$ through g.

For the initial instance, every type atom of $\mathrm{Lin}(D)$ is, by definition, of the form

$$p_{t_D^\Sigma(\alpha)}\big(\mathrm{red}(\alpha)\big)$$

for some $\alpha \in D$. Taking this atom itself as its realization and letting g be the identity on constants establishes the invariant for $\mathrm{Lin}(D)$.

Origin and saturation reconstruction TGDs do not produce type atoms. Hence the only rules that can extend the collection of type atoms are the complex TGDs.

Assume therefore that

$$p_{t_2}(b'_1, \ldots, b'_l)$$

is generated from

$$p_{t_1}(b_1, \ldots, b_j)$$

by a complex TGD

$$p_{t_1}(x_1, \ldots, x_j) \to \exists \mathbf{z}\, p_{t_2}(y_1, \ldots, y_l).$$

By the induction hypothesis, there exists an atom

$$\alpha'_1 \in I$$

of type t_1 realizing $p_{t_1}(b_1, \ldots, b_j)$.

By definition of a complex TGD, this rule exists because there are atoms $\alpha_1, \alpha_2 \in I$ of types t_1 and t_2, respectively, such that α_2 is produced from α_1 by an application of some TGD $\sigma \in \Sigma$.

We now apply the previous proposition. Since α_1 and α'_1 have the same Σ-type t_1, the application of σ producing α_2 can be transferred to α'_1. Hence there exists an application of σ whose frontier is mapped to the corresponding terms of α'_1 and which produces an atom

$$\alpha'_2 \in I$$

such that

$$t_D^\Sigma(\alpha'_2) = t_2.$$

We use α'_2 as the realization of the newly generated type atom.

Every argument of $p_{t_2}(b'_1, \ldots, b'_l)$ inherited from its body is mapped by g to the corresponding frontier term of α'_2. This is consistent with the already defined mapping g, by the construction of the complex TGD and by the previous proposition.

For each fresh null n' generated by the complex TGD, let n be the corresponding fresh null generated in the application of σ producing α'_2, and extend g by

$$g(n') = n.$$

Again, the equality component of t_2 guarantees that this definition is well defined. Therefore α'_2 realizes the newly generated type atom, and the invariant is preserved.

It remains to verify that every atom over $\mathcal{S}$ generated in J is mapped by g to an atom of I.

Atoms produced by origin reconstruction TGDs.

Consider a type atom

$$p_t(b_1, \ldots, b_j)$$

realized through g by an atom

$$\alpha = R(a_1, \dots, a_k) \in I,$$

where $t = (R, \sim, \mathcal{G})$. The associated origin reconstruction TGD produces

$$R\big(b_{\mathsf{rank}_\sim(1)}, \dots, b_{\mathsf{rank}_\sim(k)}\big).$$

By the invariant and the definition of the rank function,

$$g\big(b_{\mathsf{rank}_\sim(i)}\big) = a_i$$

for every $i \in \{1, \dots, k\}$. Consequently, the image under g of the reconstructed atom is exactly

$$R(a_1, \dots, a_k) = \alpha,$$

which belongs to I.

Atoms produced by saturation reconstruction TGDs.

Consider a saturation reconstruction TGD associated with a type

$$t = (R, \sim, \mathcal{G}),$$

a Datalog TGD $\sigma \in \Sigma$, and a pair

$$(\mathrm{body}(\sigma), \pi) \in \mathcal{G}.$$

Suppose it is applied to a type atom

$$p_t(b_1, \dots, b_j).$$

By the invariant, there exists an atom

$$\alpha \in I$$

of type t realizing this type atom.

Since

$$(\mathrm{body}(\sigma), \pi) \in \mathcal{G},$$

the definition of the Σ-type of α guarantees the existence of a homomorphism

$$\mu : \mathrm{body}(\sigma) \longrightarrow I$$

which maps every term in the domain of π to the corresponding position of α.

Since σ is Datalog, every variable occurring in $\mathrm{head}(\sigma)$ is a frontier variable. Hence, by construction of the saturation reconstruction TGD, applying g to its head gives exactly

$$\mu(\mathrm{head}(\sigma)).$$

Since

$$I = \mathrm{chase}\, D\Sigma,$$

and therefore $I \models \Sigma$, we have

$$\mu(\mathrm{head}(\sigma)) \in I.$$

Thus the image under g of every atom generated by a saturation reconstruction TGD belongs to I.

We have shown that every atom of $J_{|\mathcal{S}}$ is mapped by g to an atom of I. Since all extensions of g agree with its previous values, their union defines a homomorphism

$$g : J_{|\mathcal{S}} \longrightarrow I.$$

We have therefore constructed homomorphisms

$$\text{chase}(D, \Sigma) \longrightarrow \text{chase}(\text{Lin}(D), \text{Lin}(\Sigma)_{|\mathcal{S}})$$

and

$$\text{chase}(\text{Lin}(D), \text{Lin}(\Sigma)_{|\mathcal{S}}) \longrightarrow \text{chase}(D, \Sigma).$$

Hence

$$\text{chase}(D, \Sigma) \quad \text{and} \quad \text{chase}(\text{Lin}(D), \text{Lin}(\Sigma)_{|\mathcal{S}})$$

are homomorphically equivalent. $\square$