Logical Methods in Computer Science
Volume 22, Issue 2, 2026, pp. 8:1–8:44
https://lmcs.episciences.org/

Submitted Jan. 10, 2023
Published Apr. 21, 2026

TIGHTER BOUNDS FOR QUERY ANSWERING WITH GUARDED TGDS

ANTOINE AMARILLI [a] AND MICHAEL BENEDIKT [b]

[a] Univ. Lille, Inria Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, FR;
LTCI, Télécom Paris, Institut Polytechnique de Paris, FR
e-mail address: antoine.a.amarilli@inria.fr

[b] Department of Computer Science, Oxford University, Parks Rd, Oxford OX1 3QD, UK
e-mail address: michael.benedikt@cs.ox.ac.uk

Abstract. We consider the complexity of the open-world query answering problem, where we wish to determine certain answers to conjunctive queries over incomplete datasets specified by an initial set of facts and a set of guarded TGDs. This problem has been well-studied in the literature and is decidable but with a high complexity, namely, it is 2EXPTIME complete. Further, the complexity shrinks by one exponential when the arity is fixed.

We show in this paper how we can obtain better complexity bounds when considering separately the arity of the guard atom and that of the additional atoms, called the *side signature*. Our results make use of the technique of linearizing guarded TGDs, introduced in [GMP14]. Specifically, we present a variant of the linearization process, making use of a restricted version of the chase that we recently introduced [AB22]. Our results imply that open-world query answering with guarded TGDs can be solved in EXPTIME with arbitrary-arity guard relations if we simply bound the arity of the side signature; and that the complexity drops to NP if we fix the side signature and bound the width of the dependencies.

1. Introduction

The *open-world query answering* problem (OWQA) is concerned with evaluating a query Q on an incomplete dataset I. We specify I as a set of possible completions of an initial set of facts I_0, constrained by a set of integrity constraints Σ: formally, we evaluate Q over all completions of I_0 satisfying Σ. Typically Q is a conjunctive query (CQ), equivalent to a basic SQL query; the rules Σ limit the possible completions to consider, and are typically expressed in restricted logical formalisms. Research has therefore sought to explore the tradeoff between the expressiveness of the constraint language and the complexity of the open-world query answering problem, which we simply call *query answering* for brevity.

In the last decades, one notable focus of research on query answering is the Datalog$^{\pm}$ family of constraints: in this work we study the related formalism of *tuple-generating dependencies* (TGDs). A TGD is a universally-quantified implication, where the left-hand

We are very grateful to the anonymous referees of LMCS for several rounds of detailed feedback and corrections.

side – the *body* – is a conjunction of atoms, and the right-hand side of the implication – the *head* – is a CQ. Up to rewriting TGDs, we can ensure that their right-hand side is always a single atom: these are called *single-headed* TGDs and we focus on such TGDs in this paper, though we refer back to the case of multi-headed TGDs at the end of Section 3. A TGD is *full* if its head contains no existential quantifiers. Such TGDs are sometimes called *Datalog rules*.

TGDs are a common constraint language to express that some patterns in the data must imply the existence of other patterns; it subsumes, e.g., *inclusion dependencies* (IDs) which are common in relational databases: inclusion dependencies are TGDs where the body and head contain a single atom with no repeated variables. The formalism of guarded TGDs (GTGDs) restricts TGDs by enforcing that there is one atom in the body containing all the variables used in the body: such an atom is called a *guard atom*. This ensures that GTGDs are expressible in the guarded fragment of first-order logic. Query answering is known to be decidable in 2EXPTIME for this class, and better bounds are known in some special cases: for example, when the arity of the signature is bounded, the complexity drops to EXPTIME. Similarly, it is known that the complexity of query answering drops to PSPACE for TGDs that are *linear* [JK84, LMPS15], i.e., whose body consists of a single atom. Note that linear TGDs are a special case of GTGDs, and IDs are themselves a special case of linear TGDs. See Table 1 in Section 3 for a more detailed comparison.

Our goal in this paper is to show finer bounds on query answering. Our approach to do so is to distinguish between the relations allowed for the guard atoms in TGDs and the signature used for the remaining atoms. Specifically, we allow each GTGD to have an unrestricted guard atom, but restrict the other atoms, e.g., by bounding their maximal arity, or fixing the *side signature* from which they are taken. Roughly speaking, our results show that the first limitation suffices to bring the complexity of query answering down to EXPTIME. The second limitation further lowers the complexity to NP if we also impose another restriction, namely, bounding the so-called *width* of the GTGDs. The width is the number of *exported variables* of a GTGD, where a variable is exported when it appears in both the head and the body. Formally, the main complexity results that we show in this paper are the following, which apply to GTGDs that *obey* a side signature, i.e., that only use the relations of the side signature in rule bodies except that there may be one guard atom on a non-side-signature relation (the formal definition will be given as Definition 3.2):

Result 1. *The* OWQA *problem with GTGDs that obey a side signature is in* EXPTIME *if we assume that the arity of the side signature is bounded by a constant.*

Result 2. *The* OWQA *problem with GTGDs that obey a side signature is in* NP *if we assume that the side signature is fixed and that the width of the GTGDs is bounded by a constant.*

Note how the first result provides a unifying language to recapture the EXPTIME membership of query answering with guarded TGDs of bounded arity, and the lower complexity of linear TGDs (though we do not achieve a PSPACE but EXPTIME bound). That result is also tight in the sense that there are GTGDs with a fixed side signature of arity 1 for which OWQA is EXPTIME-complete [BBB13].

The finer-grained bounds presented in this paper were first announced in the context of our earlier work on *access patterns* [AB18], where we want to evaluate a query by accessing the underlying relations in a limited way. Research on access patterns typically relies on encoding problems with GTGDs that capture the semantics of access methods. For each

access method, the resulting TGDs will have a body consisting of a single high-arity guard atom, and multiple unary atoms over a particular predicate accessible, intuitively denoting that some instance element can be retrieved using the access patterns. Specifically, the conference version of our work [AB18] claimed a number of bounds on query answering with access patterns, using side-signature-based techniques. These results were only sketched in the conference version (for which unpublished appendixes claim a weaker form of our main result, Theorem 3.6). These results were omitted from the later journal version [AB22], which only contain bounds shown in the restricted setting of access patterns. Similar techniques are also used in a later conference paper [BBJT19] about data disclosure, by the second author and other authors.

The goal of the present work is to fill this gap, and give a self-contained presentation of our complexity results on query answering for GTGDs with a side signature. This makes it possible to understand these TGD complexity results independently from the application to access methods that motivated it. Indeed, we believe these results to be interesting in their own right, and hope that they can be useful to show complexity bounds on query answering in different contexts.

Revisiting linearization. Our key proof technique is that of *linearization*, introduced in [GMP14]. The idea is to reduce to the setting of linear TGDs, recalling that query answering for this class is known to be quite well-behaved [JK84]: it is in PSPACE in general, and in NP when the width is bounded. Restricting the side signature intuitively makes GTGDs "translatable" to linear TGDs in an auxiliary signature, where a body atom in the auxiliary signature represents a guard atom in the original signature along with a specific choice of additional atoms from the restricted side signature.

While this basic idea is quite natural, the linearization process turns out to be rather involved. In particular, before we linearize we need to pre-process the TGDs so they behave more like linear TGDs. The key part of this is a *saturation procedure* which generates derived GTGDs from our original set. Once this saturation is applied, the process of reasoning with the GTGDs becomes very similar to reasoning with linear TGDs. The saturation process is similar to ones that have been investigated in Description Logics [Bor06] and guarded TGDs [BBG$^+$24] for determining whether a ground fact is entailed (also known as "atomic query answering"). Indeed, in the process of giving our results, we will explain how linearization is a natural way to extend saturation procedures to deal with conjunctive query answering.

The main technical challenge is to use the side signature carefully to obtain a saturation whose size is not too large. In addition to the side-signature-aware construction of the saturation, we use a number of further ideas. To argue that the new saturation is correct, we must make use of a specialized version of the *chase procedure* [MMS79, One13], which augments an instance to create new entailed facts. The idea is that the saturation procedure iteratively combines multiple TGDs to get new full TGDs, and in doing so it mimics the way facts would propagate when performing a chase. But our restricted saturation only mimics a restricted flow of facts, and we must argue that this suffices. We require that it is sufficient to simulate a restricted kind of chase called the *one-pass chase*. The one-pass chase is introduced as a tool for reasoning about completeness of saturation procedures in [BBG$^+$22]. But here, again, we will need to customize it to the side signature-aware context. See Section 5 for the formal definitions.

Our linearization algorithm will be fairly straightforward once we close under the saturation process. But to argue that it is correct we will need yet another variation of the chase procedure. The chase with linear TGDs has a special form where one does not perform propagation of facts up and down the tree. We need to argue that GTGDs also support a variation of the chase where propagation is very restricted, the *shortcut chase*. See Section 8 for the precise definition.

Paper structure. We first give preliminaries in Section 2, reviewing in particular the notion of *semi-width* from [AB22]. We then define the notion of side signature in Section 3 and state our main result (Theorem 3.6), which describes the complexity of translating GTGDs to linear TGDs of bounded semi-width, depending on bounds on the side signature and the width of the GTGDs. Together with the results on semi-width, this result directly implies Results 1 and 2 above. The rest of the paper is devoted to giving the proof of Theorem 3.6. We start by giving some normalizations of our GTGDs (Section 4) and chase proofs (Section 5) that will be useful. We give a proof overview of the main result in Section 6, and the proof itself spans Sections 7, 8, and 9. We conclude in Section 10.

2. Preliminaries

Data and queries. We consider a *relational signature* $\mathcal{S}$ that consists of a set of *relations* with an associated *arity* (a positive integer). The *arity* of $\mathcal{S}$ is the maximal arity of a relation in $\mathcal{S}$. The set of *positions* of a relation R of $\mathcal{S}$ is the set $\{R[1] \dots R[n]\}$ where n is the arity of R. An *instance* of R is a (finite or infinite) set of n-tuples of values from some infinite set of values; we also call these values the *domain elements*, or simply *elements*. Note that the domain elements will also include *nulls*, which are the fresh values created in the facts added when performing *chase steps* with non-full tuple-generating dependencies (this will be formally defined in Section 5). An *instance* I of $\mathcal{S}$ consists of instances for each relation of $\mathcal{S}$. The *active domain* $\mathsf{Adom}(I)$ of I is the set of the domain elements that occur in tuples of I.

A *$\mathcal{S}$-atom*, or simply *atom*, is an expression of the form $R(x_1 \dots x_n)$, where R is a relation of $\mathcal{S}$ and n is the arity of R in $\mathcal{S}$. We also call $R(x_1 \dots x_n)$ an *R-atom*. We will be concerned mostly with two kinds of atoms, depending on the nature of the terms $x_1, \dots, x_n$: those where all the terms are variables, and those where all the terms are domain elements. We call the latter a *ground atom* or a *fact*. For a fact $F = R(\vec{a})$, we write $\mathsf{Adom}(F)$ for the set of elements that occur in F (i.e., those of $\vec{a}$), and we also call F an *R-fact*. We will equivalently see instances I as a set of *facts* $R(a_1 \dots a_n)$ for each tuple $(a_1 \dots a_n)$ in the instance of each relation R. A *subinstance* I' of I is then an instance that contains a subset of the facts of I: we equivalently say that I is a *superinstance* of I'.

A *homomorphism* from a set of atoms $A_1, \dots, A_n$ to a set of atoms $B_1, \dots, B_m$ is a substitution σ of the variables occurring in the A_i such that, for each atom A_i, writing $\sigma(A_i)$ the result of substituting its elements according to σ, then the result is one of the $B_1, \dots, B_m$.

We will study *Boolean conjunctive queries* (CQs), which are logical expressions of the form $\exists x_1 \dots x_k \, (A_1 \wedge \dots \wedge A_m)$, where the A_i are atoms over $x_1 \dots x_k$. Note that we only focus on *Boolean CQs*, which have no free variables. For this reason, throughout the paper, *by default when we say a CQ we mean a Boolean one*. Also note that we do not allow constants in CQs, but these can be encoded, e.g., by expanding the signature with additional unary predicates to distinguish the constants in instances. We further discuss the impact of constants in Section 3. A *match* of a CQ Q in an instance I is a homomorphism from Q

to I, i.e., a mapping h from the variables of Q to $\mathsf{Adom}(I)$ which ensures that, for every atom $R(t_1 \dots t_n)$ in Q, we have that $R(h(t_1) \dots h(t_n))$ is a fact of I. We say that Q *holds* in I if there is a match of Q in I.

Integrity constraints. We study *integrity constraints* which are defined in (fragments of) first-order logic (FO), disallowing constants. In our definition of FO, we only consider FO formulas and FO fragments where all quantified variables that appear in a formula must appear in some relation of the formula. This ensures that the satisfaction of a constraint on an instance I only depends on the active domain, i.e., on the values occurring in facts of I; in other words, we follow the active-domain semantics. For an FO formula ρ and an instance I, we say that I *satisfies* ρ, written $I \models \rho$, if ρ holds on I with the usual semantics of FO; we omit the corresponding definitions (see, e.g., [Lib95]). Otherwise, we say that I *violates* ρ, written $I \not\models \rho$. For Σ a set of FO formulas, we write $I \models \Sigma$ to say that I satisfies all formulas of Σ.

We focus on a specific fragment of FO, called *tuple-generating dependencies* (TGDs), which we now review. A *tuple-generating dependency* (TGD) is an FO sentence γ of the form: $\forall \vec{x}\ (\beta(\vec{x}) \rightarrow \exists \vec{y}\ A(\vec{x}, \vec{y}))$ where β is a conjunction of atoms called the *body* and where all variables from $\vec{x}$ appear, and A is an atom called the *head*. We require that all variables of $\vec{x}$ occur in β and that all variables of $\vec{y}$ occur in $A(\vec{x}, \vec{y})$, but $A(\vec{x}, \vec{y})$ may only use a subset of the variables of $\vec{x}$ (or possibly none at all). Note that we define here *single-headed* TGDs, whose head consists of a single atom. *We focus on single-headed TGDs throughout this paper*: see for instance [GMP20] for an example of work investigating the impact of this choice, and see the end of Section 3 for a further discussion of the matter.

For γ a TGD and I an instance, using the previous notation, a substitution τ from β to I is called a *trigger* of γ in I. The trigger τ is said to be *active* if there is no mapping τ' that extends τ and ensures that $\tau'(A(\vec{x}, \vec{y})$ occurs in I. The semantics of γ is that I satisfies γ if and only if there are no active triggers of γ in I.

For brevity, in the sequel, we will *omit outermost universal quantifications in TGDs*. The *exported variables* of γ (also called *frontier variables*) are the variables of $\vec{x}$ which occur in the head. A *full TGD* is one with no existential quantifiers in the head. A *guarded TGD* (GTGD) is a TGD γ whose body β contains some atom A which contains all variables occurring in β. We call A a *guard* of β, and of γ: note that it is not necessarily unique. When decomposing β as $A(\vec{x}) \wedge \beta'(\vec{x})$ for one specific choice of A, we call A the *guard atom*.

We will also say that a fact or set of facts S is *guarded* by another fact or set of facts S' if the facts of S only use values occurring in S'.

A *linear TGD* is a TGD where the body consist of a single atom: remember that we defined TGDs to be single-headed so we already know that the head is also a single atom. Also notice that a linear TGD is always guarded. An *inclusion dependency* (ID) is a linear TGD where we further impose that no variable is repeated in the body and no variable is repeated in the head.

The *width* of a TGD is the number of exported variables. Remember that we do not allow constants in TGDs.

Fact entailment, OWQA, and TGD entailment problems. We refer throughout the paper to the standard notion of entailment in first-order logic:

Definition 2.1. We say that a set of FO sentences λ *entails* an FO sentence ρ, written $\lambda \models \rho$, if every instance satisfying λ also satisfies ρ.

In particular, a special case of entailment is *entailment of a TGD τ by a set of TGDs* Σ, which we will use in the technical proofs. Another case, which is the main focus of this paper, is that of entailment problems of the form:

$$\bigwedge_{i \leq n} F_i \wedge \Sigma \models Q$$

where Σ is a set of TGDs, each F_i is a fact, and Q is a CQ. This is the problem of *certain answers* or *(open-world) query answering* [FKMP05] (OWQA) under TGDs for CQs.

Definition 2.2. If I_0 is a finite instance with facts $F_1 \dots F_n$, we also write $I_0, \Sigma \models Q$ to mean $\bigwedge_i F_i \wedge \Sigma \models Q$.

The OWQA *problem* is the problem of deciding whether such entailments hold. Formally, the input to OWQA consists of a finite instance I_0, a set Σ of TGDs, and a CQ Q; the output is a Boolean indicating whether $I_0, \Sigma \models Q$ or $I_0, \Sigma \not\models Q$.

One variant of OWQA that we will study is *fact entailment*. In this problem, the input consists of a finite instance I_0, a set Σ of TGDs, and a fact F on the domain of I_0. The output is a Boolean indicating whether $I_0, \Sigma \models F$, i.e., the fact F is contained in every superinstance of I_0 that satisfies Σ.

Note that the OWQA problem is closely connected to the problem of *query containment under constraints*, which has been independently studied in the literature, and also in connection with query containment under access patterns; see, e.g., [DLN07].

In this paper, we study the complexity of the OWQA problem in *combined complexity*, i.e., as a function of $|I_0|$, $|\Sigma|$, and $|Q|$, where the size $|\Sigma|$ is taken following, e.g., a string representation, and likewise for $|I_0|$ and $|Q|$.

We now discuss two ways in which OWQA problems can be equivalent, while possibly changing the underlying signature. One first notion is *entailment-equivalence*, which is defined on sets of TGDs:

Definition 2.3. For $\mathcal{S}$ a signature, we say that two finite sets of TGDs Σ and Σ' are *$\mathcal{S}$-entailment-equivalent* if they are interchangeable for OWQA on $\mathcal{S}$, namely: for any instance I_0 over $\mathcal{S}$ and CQ Q over $\mathcal{S}$, we have $I_0, \Sigma \models Q$ iff $I_0, \Sigma' \models Q$.

Note that Σ and Σ' may be on a larger signature than $\mathcal{S}$.

Remark 2.4. Note that entailment-equivalence is weaker than logical equivalence. Indeed, if Σ and Σ' are logically equivalent in the sense that $\Sigma \models \Sigma'$ and vice-versa, then they are $\mathcal{S}$-entailment-equivalent over any signature $\mathcal{S}$. However, the converse is not true: for ϕ the GTGD $R(x) \rightarrow S(x)$, letting $\Sigma = \{\phi\}$ and $\Sigma' = \emptyset$, and $\mathcal{S}$ be a signature containing R but not S, then Σ and Σ' are not logically equivalent but they are $\mathcal{S}$-entailment-equivalent.

A second notion is *emulation*, which is defined on pairs of a set of TGDs and of an instance:

Definition 2.5. For $\mathcal{S}$ a signature which is a subsignature of $\mathcal{S}'$, we say that a set Σ' of constraints and an instance I'_0 on signature $\mathcal{S}'$ *emulates* another set of constraints Σ and instance I_0 on signature $\mathcal{S}$ if they are equivalent for OWQA, i.e., entail the same CQs on $\mathcal{S}$. Formally, I'_0 and Σ' *emulate* I_0 and Σ if, for any CQ Q posed over the signature $\mathcal{S}$, we have $I_0, \Sigma \models Q$ iff $I'_0, \Sigma' \models Q$.

Semi-width. Our tractability results in this paper will be shown by reducing to the case of linear TGDs, which we call *linearization*. In particular, we will rely on NP upper bounds which depend on the *width* of the dependencies. The following result was shown by Johnson and Klug [JK84] in the case of inclusion dependencies:

Proposition 2.6 [JK84]**.** *For any fixed $w \geq 0$, there is an* NP *algorithm for* OWQA *under inclusion dependencies of width at most w.*

This notion was slightly generalized in [AB22] to the notion of *semi-width*, which extends width by the addition of acyclic TGDs. We now review the definition. The *basic position graph* of a set of TGDs Σ is the directed graph whose nodes are the positions of relations in Σ, with an edge from $R[i]$ to $S[j]$ if and only if there is a rule $\delta \in \Sigma$ with exported variable x occurring at position i of an R-atom in the body of δ and at position j of an S-atom in the head of δ. Note that our basic position graph is different from the notion of position graph used to ensure decidability of termination for general TGDs [One13]. We apply the basic position graph only to complexity considerations concerning linear TGDs. In particular, existentially quantified variables do not contribute to the basic position graph.

We say that a collection of TGDs Σ has *semi-width* bounded by w if Σ can be partitioned into $\Sigma_1 \cup \Sigma_2$ where Σ_1 has width bounded by w and where the basic position graph of Σ_2 is acyclic.

We can then show that OWQA is in NP for linear TGDs of bounded semi-width. The result was shown in [AB22, Appendix C] for inclusion dependencies [AB22, Proposition 6.5], and we must slightly modify the proof to apply more generally to linear TGDs:

Proposition 2.7. *For fixed w, there is an* NP *algorithm for OWQA under linear TGDs of semi-width at most w.*

To avoid distracting from the main results of the paper, we give a self-contained proof of this result in Appendix A, most of which is identical to [AB22, Appendix C].

3. Result statements

Having defined the preliminaries, we are now ready to formally state the results of this paper. We first position our work relative to existing results. Then, we introduce the notion of *side signature* through which our results are phrased. We then give the formal statement of the results. After this we state our main technical linearization result, and explain how our main results follow from that result. We finally discuss the issues of multi-headed GTGDs and constants, and explain the relationship with earlier techniques.

Earlier results. Our focus is on complexity bounds for the OWQA problem with guarded TGDs. The following bounds on this problem represent the prior state of the art [CGK13]:

Theorem 3.1 [CGK13, JK84]**.** *Given a set of guarded TGDs Σ, an instance I, and a query Q, the* OWQA *problem for Σ, I, and Q is* 2EXPTIME*-complete.*

Further, if we fix the arity of the signature $\mathcal{S}$, then the problem is EXPTIME*-complete. Last, if we fix the guarded TGDs Σ, the problem is* NP*-complete.*

If the TGDs are linear, the problem is PSPACE*-complete, and* NP *when the width is bounded. This was proven in* [JK84] *only for the case of Inclusion Dependencies, but the proof extends straightforwardly to linear TGDs.*

GTGD restriction	OWQA **complexity**
General	2EXPTIME [CGK13]
Fixed arity	EXPTIME [CGK13]
Fixed side signature arity	EXPTIME (Result 1)
Linear	PSPACE [JK84]
Fixed GTGDs	NP [CGK13]
Fixed-width IDs	NP [JK84]
Fixed side signature and width	NP (Result 2)

Table 1: Complexity results, with new results in bold

Our goal in this paper is to refine the EXPTIME and NP bounds by showing that they hold for more general constraint languages, defined via the notion of *side signature*.

The current state of the art and contribution are summarized in Table 1.

Side signature. We now introduce the notion of *side signature* that is fundamental to the statement of our results. The side signature intuitively consists of relations that can be used together with a guard in rule bodies. More precisely, we consider GTGDs where, for some choice of guard atom, all other atoms must use relations in the side signature:

Definition 3.2. Let γ be a GTGD on signature $\mathcal{S}$. Given a subsignature $\mathcal{S}' \subseteq \mathcal{S}$, we say that γ *obeys side signature* $\mathcal{S}'$ if there is a choice of guard atom in the body of γ such that all other body atoms are relations of $\mathcal{S}'$. A set of GTGDs Σ *obeys side signature* $\mathcal{S}'$ if all GTGDs of Σ do.

Definition 3.3. We refer to the relations in $\mathcal{S} \setminus \mathcal{S}'$ as *principal relations*, as opposed to *side relations*. We similarly refer to *principal atoms* and *principal facts*, versus *side atoms* and *side facts*, depending on whether the relation involved is a side relation or a principal relation. We also say that a TGD is a *principal TGD* if its head atom is a principal atom, and a *side TGD* otherwise.

In particular, a linear TGD always obeys any choice of side signature. Note that we can easily test whether GTGDs Σ obey side signature $\mathcal{S}'$: consider the body of each GTGD and check that it has at most one atom which is not in $\mathcal{S}'$ and that this atom guards the body. The choice of side signature is also not canonical, e.g., taking $\mathcal{S}' := \mathcal{S}$ always satisfies the definition, but this trivial choice will not be useful to achieve good complexity bounds using our results.

We give an example of the notion of side signature:

Example 3.4. Consider the following set of GTGDs over relations $\{R, S, T, U\}$.

$$R(x, y, x, z), T(x), T(z), U(x, z) \rightarrow \exists w\ S(y, w)$$
$$U(x, y), U(x, x) \rightarrow U(y, y)$$
$$S(x, y), U(x, y) \rightarrow S(y, x)$$

They obey the side signature $\{T, U\}$. Note that obeying a side signature does not impose constraints on head atoms, which can be principal relations or side relations. Further, when a GTGD has a body that only uses relations from a given side signature, then it obeys that side signature.

Our results will apply to sets of GTGDs Σ that obey a side signature $\mathcal{S}'$ and where other conditions are respected, in particular the arity of $\mathcal{S}'$ will have to be bounded.

Main results. We can now restate our two main results from the introduction. We show the following, which gives a sufficient condition for OWQA with GTGDs to be in EXPTIME:

Result 1. *The* OWQA *problem with GTGDs that obey a side signature is in* EXPTIME *if we assume that the arity of the side signature is bounded by a constant.*

In other words, we show that, for the OWQA problem to be in EXPTIME, we can allow arbitrary-arity guarded TGDs, and we just need to bound the arity of the side signature. In particular, our result recaptures and extends the lower complexity of OWQA for linear TGDs [JK84], except that it shows an EXPTIME bound rather than a PSPACE bound.

We further show that, once the side signature is fixed, then to achieve NP complexity, we do not need to fix the dependencies or even the arity – it suffices to fix the side signature (including its arity) and the *width* of the dependencies:

Result 2. *The* OWQA *problem with GTGDs that obey a side signature is in* NP *if we assume that the side signature is fixed and that the width of the GTGDs is bounded by a constant.*

Our result extends the NP upper bound on OWQA for bounded-width IDs shown by Johnson and Klug [JK84], as well as the NP bound with fixed GTGDs [CGK13].

Example 3.5. We give an example from the setting of *access patterns* which was mentioned in the introduction, explaining how Result 2 generalizes results proven in [AB22].

Fix a number m, consider a signature that includes a distinguished unary relation $\mathsf{accessible}$, and consider a set of TGDs of one of the two forms:

$$R(x_1 \dots x_m, \vec{y}) \rightarrow \exists \vec{z}\ H(x_{m_1} \dots x_{m_k}, \vec{z})$$

$$R(x_1 \dots x_n) \wedge \bigwedge_{i \in S} \mathsf{accessible}(x_i) \rightarrow \mathsf{accessible}(x_j)$$

In the first form of TGDs, $m_1 \dots m_k$ are numbers bounded by m. These are *linear TGDs* of *width at most* m. In isolation, query answering is known to be NP for such TGDs by a variation of [JK84].

In TGDs of the second form, j is a number in $\{1 \dots n\}$, and S is an arbitrary set of such numbers. They are *full TGDs*, but they are not linear. These are referred to as *accessibility axioms* in [AB18, AB22].

Result 2 implies that query answering for the class of TGDs of this form is in NP. Intuitively, this is because the side signature is fixed and the width of TGDs is bounded. This result is claimed (in the restricted case of inclusion dependencies, but in the broader context of result-bounded interfaces) in [AB22, Theorem 6.4]. One motivation for side signatures is to generalize this result but replacing $\mathsf{accessible}(x)$ by an arbitrary fixed side signature, giving Result 2.

Linearization result. All of our complexity bounds are shown by reducing OWQA with GTGDs to OWQA with linear TGDs. We do this using our main technical result:

Theorem 3.6. *Let $a' \in \mathbb{N}$ be a fixed bound on the side signature arity. There are polynomials P_1 and P_2 depending only on a' and an algorithm with the following input:*

- *A signature $\mathcal{S}$, where we let a be the arity of $\mathcal{S}$;*

- *A subsignature $\mathcal{S}' \subseteq \mathcal{S}$ of arity a', where we let n' be the number of relations of $\mathcal{S}'$;*
- *An instance I_0 of $\mathcal{S}$;*
- *A set Σ of GTGDs obeying side signature $\mathcal{S}'$, where we let w be the maximal width of a GTGD of Σ and let $w' := \max(a', w)$.*

The algorithm computes in time $P_1(|\Sigma| \times a \times |I_0|)^{P_2(w',n')}$ a set Σ' of linear TGDs of semi-width $\leq w'$ and arity $\leq a$, and an instance I_0^{Lin}, such that Σ' and I_0^{Lin} emulate Σ and I_0 on signature $\mathcal{S}$.

Theorem 3.6 will be proven in the next sections. Before this, we show how Theorem 3.6 allows us to prove Result 1 and Result 2.

Proving Result 1 from Theorem 3.6. Recall the statement of Result 1:

Result 1. *The* OWQA *problem with GTGDs that obey a side signature is in* EXPTIME *if we assume that the arity of the side signature is bounded by a constant.*

Proof of Result 1. We apply the reduction of Theorem 3.6, which computes in EXPTIME a finite set of linear TGDs Σ' and rewritten instance I_0^{Lin} that emulate the original GTGDs Σ and instance I_0. We can solve the OWQA problem for linear TGDs via an algorithm that takes EXPTIME in the maximum arity, with the running time of the algorithm being bounded by a polynomial in the input instance and the TGDs. This can be done either by the chase-based argument of [JK84] or via an algorithm that iteratively performs query-rewriting to generate a derived UCQ and then evaluate it, where each CQ has polynomial size in the input CQ and maximal arity of the linear TGDs. Note that the number of such CQs is polynomial in the number of linear TGDs and exponential in the arity [CLR03]. When we apply this to our exponential set of linear TGDs, we get the desired EXPTIME bound. □

Recall now the statement of Result 2:

Result 2. *The* OWQA *problem with GTGDs that obey a side signature is in* NP *if we assume that the side signature is fixed and that the width of the GTGDs is bounded by a constant.*

Proof of Result 2. We apply the reduction of Theorem 3.6 to obtain a finite set of linear TGDs Σ' and rewritten instance I_0^{Lin} that emulate the original GTGDs Σ and instance I_0, and under our assumptions the running time bound of the theorem is in PTIME because w' and n' are constant. Then, we conclude directly by Proposition 2.7. □

Constants, multi-headed dependencies, and IDs. Note that we have stated our results with dependencies that are *single-headed*, and which do not feature *constants*. We have also assumed that the query is Boolean — though non-Boolean queries can be encoded instead as a Boolean query where the output variables are instantiated with constants.

For the EXPTIME upper bound (Result 1), we do not expect that these restrictions make a difference. Intuitively, constants can be emulated by adding unary relations to the side signature (so without increasing its arity), and multi-headed rules can be rewritten to be single-headed. See Appendix B for details about this process. Our precise claim is that the result can be extended to multi-headed GTGDs with constants that may be present in the query and in rule bodies – we leave open the case of GTGDs featuring constants in rule heads.

For the NP upper bound (Result 2), rewriting constants and multi-headed dependencies may increase the width and the side signature, and we do not know if the bound still holds if we allow constants or allow multi-headed dependencies.

Inapplicability of hardness results. We also give a short explanation of why our results do not contradict the known hardness results of [CGK13] on OWQA with GTGDs.

For Result 1, it is shown in [CGK13, Theorem 6.2] that the OWQA problem on a fixed instance for an atomic query under GTGDs is 2EXPTIME-hard when the arity is unbounded, even when the number of relations in the signature is bounded. The proof works by devising a set of GTGDs that simulates an EXPSPACE alternating Turing machine, by coding the state of the Turing machine as facts: specifically, a fact $zero(\mathbf{V}, X)$ codes that there is a zero in the cell indexed by the binary vector $\mathbf{V}$ in configuration X. The arity of such relations is unbounded, so they cannot be part of the side signature $\mathcal{S}'$. However, in the simulation of the Turing machine, the GTGDs in the proof use another relation as guard (the g relation), and the bodies contain other high-arity relations. Thus, there is no choice of side signature of bounded arity which is obeyed by the set of GTGDs defined in the hardness proof of [CGK13].

For Result 2, the proof in [CGK13, Theorem 6.2] explicitly writes the state of the i^{th} tape cell of a configuration X as, e.g., $zero_i(X)$. These relations occur in rule bodies where they are not guards, but as Result 2 assumes that the side signature is fixed, they cannot be part of the side signature. A variant of the construction of the proof (to show EXPTIME-hardness on an unbounded signature arity) would be to code configurations as tuples of elements $X_1 \dots X_n$ and write, e.g., $zero(X_i)$. However, the constant-width bound on GTGDs would then mean that the proof construction can only look at a constant number of cells when creating one configuration from the previous one.

Roadmap. The rest of this paper is devoted to proving Theorem 3.6, from which we already explained how to derive our main results (Result 1 and 2). We first present two tools: a normalization of GTGDs that obey a side signature (in Section 4) and a chase process (in Section 5) which generalizes the "one-pass chase" from earlier work [BBG$^+$24] to be aware of the side signature. With these tools in place, we give a proof overview in Section 6. The proof proper is spread out over Sections 7, 8, and 9. We conclude in Section 10.

4. Simplifying GTGDs obeying a side-signature

In this section, we show a way to simplify sets of GTGDs that obey a side signature, to enforce three restrictions over them which will simplify subsequent proofs.

The first restriction that we will want to enforce is *homomorphism-closure*, which will intuitively save us from having to think about how two different exported variables may be mapped to the same element when firing a chase step:

Definition 4.1. Given a GTGD σ and a function h from the exported variables of σ to the exported variables of σ, we call $h(\sigma)$ the GTGD produced by applying h to every exported variable. A set Σ of GTGDs is said to be *homomorphism-closed* if for any GTGD δ in Σ, and mapping h as above, $h(\delta)$ is in Σ.

Note that $h(\sigma)$ is a logical consequence of σ for any function h on the exported variables.

The second restriction that we will want to enforce is that GTGDs have precisely one *principal guard*:

Definition 4.2. Let γ be a GTGD obeying side signature $\mathcal{S}'$. A *principal guard* of γ is a guard atom of the body of γ whose relation is a principal relation, recalling that this means a relation not in $\mathcal{S}'$.

Note that GTGDs obeying the side signature always have a principal guard if their body contains a principal atom, because this atom can then be used as a principal guard and the GTGD body cannot contain any other principal atom by definition of obeying a side signature. The issue is that we may have GTGDs whose body does not contain principal atoms at all. We will enforce the second restriction by rewriting the GTGDs to enforce that all GTGD bodies contain exactly one principal atom, and hence a principal guard.

The third restriction applies to non-full GTGDs: we want to make sure that every non-full GTGD is a principal GTGD, i.e., its head atom is a principal atom (Definition 3.3).

We can now define the normal form that we want to ensure:

Definition 4.3. Let $\mathcal{S}'$ be a subsignature. We say that a set of GTGDs Σ *strongly obeys* side signature $\mathcal{S}'$ if it obeys $\mathcal{S}'$ and further:

- Σ is homomorphism-closed
- Every GTGD of Σ has exactly one principal guard.
- For every non-full GTGD of Σ, the head atom is a principal atom.

Our goal in this section is to show the following result:

Proposition 4.4. *Let $\mathcal{S}$ be the signature, and let $\mathcal{S}'$ be a subsignature of $\mathcal{S}$. Let Σ be a set of GTGDs that obeys $\mathcal{S}'$, and let I_0 be an instance over $\mathcal{S}$. Let w be the width of Σ, and a' the arity of $\mathcal{S}'$.*

We can compute in time polynomial in $|I_0|$, $|\Sigma|$, and 2^w a signature $\mathcal{S}'' \supseteq \mathcal{S}$, a set Σ' of GTGDs over $\mathcal{S}''$, and an instance I'_0 over $\mathcal{S}''$, such that:

- *$|\mathcal{S}''|$ is polynomial in $|\mathcal{S}|$ and $|\Sigma|$;*
- *The width of Σ' is at most $\max(a', w)$;*
- *Σ' strongly obeys $\mathcal{S}'$;*
- *I'_0, Σ' emulates I_0, Σ on signature $\mathcal{S}$.*

Thanks to this result, towards showing Theorem 3.6, we can first apply the result and assume that the input GTGDs Σ strongly obey side signature $\mathcal{S}'$.

We prove Proposition 4.4 in the rest of this section, in successive steps.

4.1. **Principal guards and principal head atoms.** The first step to prove Proposition 4.4 is to enforce the condition on principal guards, which will also incidentally imply the condition on non-full GTGDs. This is the only step that will modify the input instance I_0.

Some GTGDs of Σ have a body already featuring an atom on a principal relation, in which case they already have a principal guard, which is unique because Σ obeys $\mathcal{S}'$. However, other GTGDs of Σ do not have any atom on a principal relation in their body. To ensure that such GTGDs have a principal atom, we will add new principal relations that can be used as principal guards.

Let us expand the signature $\mathcal{S}$ to $\mathcal{S}''$ by creating, for each side relation R in $\mathcal{S}'$, a new principal relation R' in $\mathcal{S}'' \setminus \mathcal{S}'$ with same arity as R. Let us modify the GTGDs of Σ as follows: for each side relation R, in every GTGD of Σ with an R-atom in the head, replace it with an R'-atom on the same variables. Further, for each side relation R, let us add to Σ' the full GTGD $R'(\vec{x}) \to R(\vec{x})$. Last, in every GTGD γ of Σ which does not have a

principal guard, pick a guard atom A on some side relation R, and replace γ in Σ' by a rule γ' obtained from γ by adding to the body of γ an atom A' on the same elements as A with the principal relation R', which will serve as principal guard.

Lastly, to rewrite the instance I_0 to I'_0, we do the following: for each fact $R(\vec{a})$ on a side relation R, we add the fact $R'(\vec{a})$.

We first claim that the transformation is correct, namely:

Claim 4.5. I'_0, Σ' emulates I_0, Σ on signature $\mathcal{S}$.

Proof. Assume first that a query Q on $\mathcal{S}$ is not entailed by I_0, Σ, i.e., there is a counterexample superinstance I of I_0 which satisfies Σ and does not satisfy Q. Then we build I' by adding the fact $R'(\vec{a})$ for every side fact $R(\vec{a})$ of I. The query Q is still not satisfied by I' because the restriction of I and I' to relations of $\mathcal{S}$ is identical. Further, I' is a superinstance of I'_0, and it is easy to see that I' satisfies Σ' because I satisfies Σ.

For the converse direction, let Q be a query which is not entailed by I'_0, Σ', let I' be a counterexample model, and build I from I' by removing all facts of $\mathcal{S}'' \setminus \mathcal{S}$. Then I is a superinstance of I_0 that does not satisfy Q. To see why I satisfies Σ, let γ be a GTGD of Σ and let τ be a trigger of γ in I. There is a corresponding GTGD γ' in Σ' obtained by possibly adding one principal guard atom, and possibly changing the head. We claim that τ is also a trigger of γ' in Σ'. Indeed, in I', for every side fact $R(\vec{a})$, the fact $R'(\vec{a})$ also exists. This is by construction of I'_0 for the facts of I'_0, and for the other facts of I' it is because Σ' ensures that facts $R(\vec{a})$ can only be derived from the GTGD $R'(\vec{x}) \to R(\vec{x})$. Thus, τ is also a trigger of γ', i.e., the possibly extra atom in the body of γ' is also mapped by τ. Thus, as I' satisfies Σ', and together with rules of the form $R'(\vec{x}) \to R(\vec{x})$, we know that the head of γ also exists in I. □

Now, the resulting Σ' still obeys the side signature. Further, the new signature $\mathcal{S}''$ is such that $|\mathcal{S}''| \leq 2|\mathcal{S}|$, and the process is polynomial in $|\Sigma|$ and in $|I_0|$. The width of GTGDs of Σ' is at most the width of the GTGDs of Σ, except we added full GTGDs (from $\mathcal{S}'' \setminus \mathcal{S}$ to $\mathcal{S}'$) whose width is a'. Thus, the result of this transformation satisfies the conditions of Proposition 4.4, and ensures that each GTGD now has a principal guard.

Note that, from the way we changed the dependencies, every non-full GTGD is a principal GTGD, i.e., has a head atom which uses a principal relation of $\mathcal{S}''$. Indeed, the only GTGDs with a $\mathcal{S}'$ atom in their head after the rewriting are the full GTGDs of the form $R'(\vec{x}) \to R(\vec{x})$ that we added, and these are full.

4.2. **Homomorphism-closure.** The second and last step is to enforce homomorphism-closure on the resulting set of GTGDs. We do so simply by considering each GTGD and every possible way to identify the exported variables and add the resulting GTGD to Σ.

This process does not affect $\mathcal{S}$-entailment-equivalence, because the resulting GTGDs are logically entailed by Σ. The resulting GTGDs also have width no greater than that of the original GTGDs. Each GTGD still has exactly one principal guard, and each non-full GTGD still has a principal atom in its head. The process runs in time polynomial in $|\Sigma|$ and in 2^w, where w is the width bound.

The resulting set of GTGDs is now homomorphism-closed and the other hypotheses are still true, so we have finished the proof of Proposition 4.4.

5. One-pass tree-like chase proofs for guarded TGDs

To show our linearization result, we will need a notion of *chase proofs*. We will more specifically use *one-pass tree-like chase proofs*: we review the result from [BBG+22] that they can be used for OWQA with GTGDs, and show a variant of this result that we will use.

Tree-like chase proofs. We first review the general notion of *chase sequences* [FKMP05], which are known to be complete for OWQA with CQs and general TGDs. We specifically focus on *tree-like chase proofs*, which are complete for OWQA with guarded TGDs [CGL12, BLM10]. We first define an abstract structure of *chase tree*, before clarifying in the rest of the section how we construct chase trees from an instance and a set of GTGDs. Our notion of chase trees also distinguishes one node in the chase tree, which is said to be *recently updated*; we will use the recently updated node later when defining one-pass chase proofs.

Definition 5.1. A *chase tree* consists of a directed tree T, a function mapping each node v in the tree to a finite set of facts written $T(v)$, and a choice of a node of T called the *recently updated* node. We abuse notation and write T to mean both the chase tree and its underlying directed tree.

Each chase tree has an *underlying instance*, which is just the union of the facts $T(v)$ over all nodes v in the tree.

We now explain how chase trees can be extended by performing *chase steps* with GTGDs and so-called *propagation steps*. These steps will then be used to define *tree-like chase sequences* and their variants. A chase tree T can be transformed to another chase tree T' in two ways:

- One can apply a *chase step* with a GTGD $\gamma : \forall\vec{x}(\beta(\vec{x}) \rightarrow \exists\vec{y}\ A(\vec{x},\vec{y}))$. Recall from the preliminaries the definition of triggers and of active triggers. Assume that we have a node v in T and a trigger τ of γ in T such that we have $\tau(\beta) \subseteq T(v)$.Then we can apply a chase step, which we also call *firing* τ (on v). It will ensure that τ is not active in the underlying instance of T, as we will add facts to T that define an extension τ' of τ with an image for the head of γ.

  The result of the chase step is obtained as follows.
  - If γ is full, then the chase tree T' is obtained from T by marking v as recently updated in T', setting $T'(v) := T(v) \cup \{\tau(A)\}$, and defining the function T' on other nodes in the same way as T.
  - If γ is not full, then τ is extended to a substitution τ' that maps each variable in $\vec{y}$ to a value not occurring in T, all these values being pairwise distinct. The fresh values are domain elements that we call *labelled nulls* or simply *nulls*. The chase tree T' is obtained from T by introducing a fresh child v' of v, marking v' as recently updated in T', and defining T' by extending T with $T(v')$ which will always contain $\tau'(A)$, and additionally will contain a subset of the following facts of $T(v)$:

    $$\{F \in T(v) \mid F \text{ is guarded by } \tau'(A)\}$$

    In other words, the new node contains $\tau'(A)$ and some facts of the parent node that are guarded by it, i.e., some facts of $T(v)$ that only use elements shared with $T(v')$. We refer to the facts other than A as *inherited facts* of the child node v'.
- One can apply a *propagation step* from a node v to a node v' in T. More precisely, we select a nonempty subset $S \subseteq T(v)$ of the facts of v, and select a node v' where these facts do not occur (i.e., $T(v') \cap S = \emptyset$) but they are guarded (i.e., the facts of S only use

elements occurring in a fact of $T(v')$). Then we set $T'(v') := T(v') \cup S$ and mark v' as recently updated.

Note that, in the definitions above, we allow the firing of non-active triggers. For a non-full TGD firing a non-active trigger means that we create a new child of v, with a different choice of fresh value for the existentially-quantified variables.

A *tree-like chase sequence* for an instance I and a finite set of GTGDs Σ is a finite sequence of chase trees $T_0, \ldots, T_n$. In the sequence, the initial chase tree T_0 consists of exactly one *root node* r, with $T_0(r) := I$, and with r being the recently updated node in T_0. Then, each T_i with $0 < i \leq n$ is obtained from T_{i-1} by one of the two steps above, i.e., a chase step with some $\gamma \in \Sigma$, or a propagation step. For each node v in T_n and each fact $F \in T_n(v)$, this sequence is a *tree-like chase proof of F from I* and Σ. It is well-known (e.g., [CGL12]) that, for any CQ Q, we have $I, \Sigma \models Q$ if and only if there is a tree-like chase sequence $T_0, \ldots, T_n$ for I and Σ such that Q has a *match* in T_n, in other words, there is a tree-like chase proof $T_0, \ldots, T_n$ of each of the facts of $\sigma(Q)$, for σ some substitution mapping the variables of Q to the domain values of T_n. Note that the facts of Q may be witnessed in different nodes, i.e., it may be the case that there is no single node v of T_n such that $\sigma(Q) \subseteq T_n(v)$.

Dating back at least to [CGLP11], it is known that entailment from GTGDs is witnessed by tree-like chase sequences. We will not make use of results from [CGLP11] directly, but rather make use of a specialized tree-like chase, based on a construction in [BBG$^+$24], which we explain below.

5.1. Restricted tree-like chase proofs.

Our linearization result will rely on the fact that tree-like chase proofs can be normalized to ensure that we do not jump back and forth in a tree, but perform our changes to the tree in one single traversal. Versions of this result have appeared dating back to [AB18], with refinements in [Kap19] for the disjunctive case. The result was applied to get rewritings for atomic queries under GTGD constraints in [BBG$^+$22], and we use the formulation from that paper.

Definition 5.2 [BBG$^+$22]**.** A tree-like chase sequence $T_0, \ldots, T_n$ for an instance I and a finite set of GTGDs Σ is *one-pass* if, for each $0 < i \leq n$, the chase tree T_i is obtained by applying one of the following two steps to the recently updated node v of T_{i-1}:

- a propagation step copying exactly one fact from v to its parent (which then becomes the recently updated node);
- a chase step on v with a GTGD from Σ (then either v stays as recently updated node or the chase step creates a child of v which becomes the recently updated node).

Further we can only do the second when the first does not apply.

Thus, each chase step in a tree-like chase sequence is applied to a "focused" node, namely, an ancestor of the node that was updated by the previous chase step. Steps with non-full TGDs move the "focus" from parent to child, and steps with full TGDs do not move the focus: the full steps are followed by propagation which copies the fact rootwards as long as possible, possibly moving the "focus" rootwards. Moreover, once a child-to-parent propagation has taken place, the child can never be revisited in further steps. Indeed, whenever a node stops being the recently updated node, then the only way it can become recently updated again is following a propagation step, which always goes rootwards. Hence,

if the parent of v node becomes the recently updated node, then the subtree rooted at v will never be revisited again and will no longer be modified.

Theorem 5.3, proven in [BBG$^+$24], shows that one-passness can be enforced on chase proofs for GTGDs for fact entailment: whenever a proof exists, there exists a one-pass proof too.

Theorem 5.3 (Theorem 4.2 of [BBG$^+$24])**.** *For each instance I, each finite set of GTGDs Σ and each fact F such that $I, \Sigma \models F$, there exists a one-pass tree-like chase proof of F from I and Σ.*

We will need a variant of this one-pass chase process that is aware of the side signature. We first liberalize the notion of tree-like chase by modifying the definition of chase steps: we want to also allow the creation of child nodes when performing a chase step with a full GTGD:

Definition 5.4. A *relaxed tree-like chase* is defined like the notion of tree-like chase defined previously: it maintains along with each instance a tree structure T_i, and designates one node of the tree structure as recently updated. The only difference is that we allow *relaxed chase steps with full TGDs*. To perform such a step, we consider a full GTGD γ having head atom A and a trigger τ for γ on some node v in the current tree T. Performing the relaxed chase step means extending T to T' by introducing a fresh child v' of v, marking v' as recently updated in T', and defining T' that extends T by setting $T(v')$ to contain the instantiation $\tau(A)$ of the head atom, along with a subset of the facts that are guarded by $\tau(A)$. That is, in the relaxed step, we do the same surgery on the tree that we would do on a non-full TGD chase step in a tree-like chase, except that no fresh values are introduced.

We then introduce the specific variation that we will use:

Definition 5.5. Let Σ be a finite set of GTGDs strongly obeying side signature $\mathcal{S}'$. A *principal-exempt one-pass chase* for an instance I and for Σ is a relaxed tree-like chase sequence $T_0, \ldots, T_n$ where for each $0 < i \leq n$, the chase tree T_i is obtained by applying one of the following three steps to the recently-updated node v of T_{i-1}:

- a propagation step copying exactly one side fact from v to its parent;
- a chase step on v with a GTGD from Σ, where the GTGD can be either a side GTGD or a non-full GTGD;
- a relaxed chase step with a full principal TGD.

Like in the one-pass chase, we further require that we can only perform a chase step (of any kind) if no propagation step (of a side fact) applies. We also restrict the facts inherited when performing chase steps (of either kind) that add a new child node: we require that only side facts can be inherited when creating new nodes.

It is easy to see that, as Σ strongly obeys the side signature, in a principal-exempt one-pass chase sequence every non-root node of every tree contains precisely one principal fact (which is added when the node is created). In particular the definition implies that principal facts are never inherited and also never propagated, so each principal fact exists in precisely one node. Intuitively, a principal fact F does not need to be inherited or propagated because triggers containing F can always be assumed to use F as a guard, and so we will show that they can always be applied on the node that contains F.

We now claim a side-signature-aware analogue of Theorem 5.3: for fact entailment, we can always assume that chase proofs are principal-exempt one-pass.

Theorem 5.6. *For each instance I_0, each finite set of GTGDs Σ obeying side signature $\mathcal{S}'$, and each fact F such that $I_0, \Sigma \models F$, there exists a principal-exempt one-pass tree-like chase proof of F from I_0 and Σ.*

Proof. We proceed by reducing to Theorem 5.3. Given the constraints Σ, we build new constraints Σ' over a modification of the signature where every principal relation R of arity n is replaced by a principal relation R' of arity $n+1$. For each rule $\sigma \in \Sigma$ we form the rule σ' over the revised signature by performing the following replacements:

- Letting $R(\vec{x})$ be the principal guard of σ, which exists because Σ strongly obeys $\mathcal{S}'$, we replace it by $R'(\vec{x}, t)$ where t is a fresh variable.
- If the head of σ is a principal atom $R(\vec{x})$, then we replace it by $R'(\vec{x}, t')$ where t' is a fresh variable that is existentially quantified.

For a fact $F = R(\vec{x}_0)$ where R is a principal relation (i.e., a principal fact), we let $F' = R'(\vec{x}_0, t_0)$ where t_0 is a fresh constant – again, a distinct one for each fact. For a side fact F, we let $F' = F$. For a finite set of facts I_0 we let I_0' be formed by applying this transformation to each fact.

We first claim the following equivalence (*): we have $I_0, \Sigma \models F$ if and only if $I_0', \Sigma' \models F'$.

In one direction, consider I extending I_0 that satisfies $\Sigma \wedge \neg F$. We form I' from I by applying the priming transformation above. It is easy to see that I' extends I_0', does not contain F', and also satisfies Σ'. For the latter, suppose we have a trigger τ' for $\gamma' \in \Sigma'$ in I'. By dropping the extra arguments we get a trigger τ for $\gamma \in \Sigma$ in I, thus we have a corresponding fact H witnessing the head in I, and thus I' witnesses the head of γ' in I'.

In the other direction, suppose we have I' extending I_0' satisfying $\Sigma' \wedge \neg F'$. We form I by simply dropping the last argument of every principal fact. It is also straightforward that I extends I_0, does not contain F, and satisfies Σ. For the latter, suppose we have a trigger τ for $\gamma \in \Sigma$ in I. By the definition of I, each principal fact can be extended in I' with an extra argument. This gives a trigger τ' for $\gamma' \in \Sigma'$ within I'. Thus, there is a fact H' witnessing the head of γ'. We form H by dropping the final argument from H' if it is a principal fact, otherwise we take $H = H'$. This is the fact required to witness that τ is not active. This establishes that I satisfies $\Sigma \wedge \neg F$ and proves the equivalence (*).

By Theorem 5.3, entailment of Q' by $I_0' \wedge \Sigma'$ is witnessed by a one-pass proof $T_1' \dots T_n'$. We modify such a proof to $T_1 \dots T_n$ by simply dropping the final argument in each fact within each node of a tree, each chase step, and each propagation step. We claim that this can be used to construct a principal-exempt proof of Q from I_0 according to Σ. The principal-exempt proof will be obtained by performing propagation steps and chase steps in the same way as in $T_1 \dots T_n$, maintaining that after each step the recently updated node in T_i is the one corresponding to the recently updated node in T_i'.

Let us explain the process more precisely. Consider the case where T_{i+1}' is formed from T_i' by applying a chase step with a non-full rule $\gamma' \in \Sigma'$ with trigger τ' on node v_i'. There are two cases: either γ' corresponds to a non-full rule γ of Σ, or it corresponds to a full rule γ of Σ with a principal atom in the head. In the first case, we can fire γ on τ in T_i on the node v_i corresponding to v_i' in T_i, inheriting the facts corresponding to the facts inherited when firing τ_i' on v_i'. In the second case, we can fire γ to perform a relaxed chase step, inheriting the facts corresponding to the facts inherited when firing τ_i' on v_i'.

Consider now the case where T_{i+1}' is formed from T_i' by applying a chase step with a full rule $\gamma' \in \Sigma'$; this rule corresponds to a full rule γ in Σ with a side fact in the head. We fire the corresponding trigger on T_i and create the same fact with a full chase step which has a

side atom in the head; and we propagate the fact as much rootwards as it is propagated in T'_i. Note that this propagates the newly created facts as much rootwards as possible, because the new fact is guarded by the same nodes in T_i and in T'_i.

In the sequence $T_1 \dots T_n$, principal facts are never propagated rootwards, because they are created by firing a trigger which in $T'_1 \dots T'_n$ is a trigger of a non-full rule that creates a fact featuring at least one fresh value: thus the fact is not propagated rootwards in $T'_1 \dots T'_n$ hence not in $T_1 \dots T_n$. Further, principal facts are never inherited in $T_1 \dots T_n$, because in $T'_1 \dots T'_n$ these facts feature a fresh value at the extra position, which is never an exported variable in any rule: so when we create a new child node in $T'_1 \dots T'_n$ then these facts are never guarded by the child node and so can never be inherited, hence the corresponding facts are also not inherited in $T_1 \dots T_n$. □

6. Proof overview of the linearization result (Theorem 3.6)

With our normalization of GTGDs and chase proofs out of the way, we can now begin the proof of Theorem 3.6. The construction that we use is a refinement of the linearization method given in Section 4.2 of Gottlob, Manna, and Pieris [GMP14]. Prior to giving the proof, we begin with an overview.

There is a pretty obvious strategy for linearization: introduce a new predicate to refine each fact by indicating the set of facts over the side signature that it guards, then rewrite the GTGDs into linear TGDs over the enhanced signature. One also would modify the initial instance accordingly. The correctness of this transformation is less evident. We discuss informally what the issue is. Entailment with guarded TGDs is captured by the chase process, which forms an instance with a tree-like shape, as we explained in Section 5. At each chase step we may add new nodes to the tree, but we may also need to propagate facts up and down the tree. With linear TGDs, we can use the same process, but *we no longer propagate facts up and down the tree*: we simply create new nodes containing a single fact per node. This is one of the big advantages of reasoning with linear TGDs over more general guarded TGDs. It could be the case that, in performing this linearization, we are losing track of the effect of some propagation steps. We need to use the notion of principal-exempt one-pass chase to justify that this may not happen (specifically, we will use Theorem 5.6). And we also need to ensure that the rules are *closed* under a form of logical derivation, to ensure that propagation is unnecessary.

Our proof strategy consists of three steps.

First, we explain how to compute a form of *saturation* of our GTGDs, which adds full GTGDs that are derived from the original set. This is a kind of rewriting result for atomic query answering (as in [BBG$^+$22]). To ensure that the process satisfies our running time bounds, we restrict it to be complete only on a limited subset of instances (the *childish* instances), and we do not add *every* derived full GTGDs but restrict to a limited set of GTGDs, called *suitable*. Because of this restriction, some care is needed to argue that the resulting saturation is "complete" – for example, that we can derive new facts on an initial childish instance by just evaluating the full TGDs in the saturation. This completeness is justified using the principal-exempt one-pass chase: we show that our saturation contains enough rules to account for the propagation in this version of the chase, which we already know is complete.

This saturation intuitively ensures that, whenever a full GTGD generates a fact about already-existing elements, then this generation could already have been performed when these earlier elements had been generated, using an implied full GTGD in the saturation. The main difference of our saturation with [GMP14] is that we exploit the width and side signature arity bounds to compute only a portion of the derived rules (the *suitable* ones), without bounding the overall signature.

Second, once this saturation has been computed, we return to an analysis of the chase and explain how to structure it further. Specifically, we enforce that we only fire full GTGDs and their bounded-breadth closure after we have fired a non-full GTGD. We call this the *shortcut chase*, since we shortcut certain derivations that go up and down the chase tree via the firing of derived rules. We have therefore achieved our intermediate goal: *a variation of the chase process for GTGDs that is complete for fact entailment, but with no propagation steps at all.*

Third, using this propagation-free shortcut chase, we can turn to linearization. We perform the linearization described informally above – introduce auxiliary predicates and then rewrite the source GTGDs to use these predicates. The shortcut chase can then be used to argue that the transformation is correct.

The rest of the paper will give the details of this proof template. We fix the side signature arity bound $a' \in \mathbb{N}$ throughout the proof. Given the input signature and subsignature, the instance, and the GTGDs, we first apply Proposition 4.4 to compute the signature $\mathcal{S}$, the side signature $\mathcal{S}'$, the instance I_0, and the GTGDs Σ which strongly obey $\mathcal{S}'$, while ensuring that I_0 and Σ emulate the original instance and GTGDs. Remember that this process is polynomial in the input except that it is exponential in the original width bound, and that the new signature is polynomial in the original signature. We let w be the maximal width of the GTGDs of Σ, which is at most the maximum of a' and of the original width bound. Up to adding trivial rules to Σ, we ensure that the maximal width w of a GTGD of Σ is such that $w \geq a'$.

7. Computing a size-controlled saturation

We start the first step of our proof by revisiting the well-known notion of *saturation* of a set of rules. Informally, a saturation of a finite set of GTGDs is a finite set of *full* GTGDs that derive the same ground consequences as the original set, and so is complete for fact entailment. The notion is closely related to the notion of saturation in resolution theorem-proving, and the notion of a *rewriting* from [BBG+24], but our formalization will be slightly different.

The saturation that we will define will not be complete in general instances, but only on specific kinds of instances, which we call *childish instances*. These instances are intuitively the ones that can be obtained as the result of performing a chase step with a bounded-width GTGD, which creates a principal fact and inherits some side facts. Formally:

Definition 7.1. Let w be the maximal width of GTGDs of Σ. A *childish instance* is an instance $\{F\} \cup I'$ consisting of one principal fact $F = R(\vec{a})$ which is an isomorphic copy of some GTGD head of Σ, together with a set I' of side facts which is guarded by F and where $\mathsf{Adom}(I')$ has cardinality at most w.

Given a finite set of GTGDs Σ, we say that a finite set of full GTGDs Σ' is *complete for fact entailment* (with Σ) if, whenever we apply it to an instance I, then all facts on the

domain of I entailed by I and Σ are derived by Σ'. A *childish saturation* is then a saturation which is complete for fact entailment on childish instances only. Let us define these notions formally:

Definition 7.2. A *childish saturation* of a finite set of GTGDs Σ is a finite set of full GTGDs Σ' such that:

- Every GTGD in Σ' is logically entailed by Σ;
- Σ' is complete for fact entailment over childish instances: for every childish instance I, every fact F entailed by I and Σ is also entailed by I and Σ', in other words, $I, \Sigma \models F$ implies $I, \Sigma' \models F$.

It is known (see, e.g., [BBG$^+$24]) that every finite set Σ of GTGDs has a finite saturation composed of GTGDs that are complete for fact entailment on arbitrary finite instances (in particular on childish instances). In fact, one suitable choice is simply to take all full GTGDs that are entailed by Σ. In this section, we show that, for dependencies that strongly obey a side signature, and over childish instances, we can compute a finite saturation that is not too large:

Theorem 7.3. *There is an algorithm taking as input a set of GTGDs Σ that strongly obeys side signature $\mathcal{S}'$ and produces a childish saturation, where the maximum width is bounded by the maximum width of Σ, and the running time is bounded by*

$$\mathrm{Poly}(|\Sigma|, a^{O(w)}, 2^{n' \times w^{a'}}).$$

where a is the maximal arity of the relations of $\mathcal{S}$, a' is the maximal arity of the relations of $\mathcal{S}'$, n' is the number of relations of $\mathcal{S}'$, and w is the maximum width of a GTGD of Σ (assumed to be no smaller than a').

We prove this theorem in the rest of this section. We will define the *$\mathcal{S}'$-suitable saturation* of Σ, and show that it has the properties required by the theorem. We will reason about full GTGDs with side signature $\mathcal{S}'$ that are *suitable*, i.e., that obey three requirements: having width at most w, being Σ-*compatible*, and satisfying a certain *breadth* restriction. The notion of Σ-*compatibility* means that the head atoms and principal guard atoms of GTGDs are *compatible* with Σ in the sense that they are isomorphic to some principal head atom of Σ. Restricting to such GTGDs is necessary to avoid considering a number of GTGDs which would be exponential in the signature arity. As for *breadth*, it bounds how many different variables can be used by the side atoms of any given GTGD, again avoiding an exponential blowup in the number of possible GTGDs as a function of the principal signature arity. The notion of breadth intuitively means that the body of a suitable full GTGD is a childish instance, up to replacing variables by domain elements. Formally:

Definition 7.4. We say an atom A is Σ-*compatible* if it is a side signature atom or if there is a head atom in a GTGD of Σ to which it is isomorphic.

Let γ be a full GTGD obeying side signature $\mathcal{S}'$ having width at most w and having a principal guard B_γ. Let H_γ be its head atom. We say that γ is Σ-*compatible* if each one of H_γ and B_γ are Σ-compatible (not necessarily with the same atom of Σ).

Definition 7.5. For any $b \in \mathbb{N}$, we say that γ has *breadth* $\leq b$ if, letting A be its principal guard, then there is a subset X of at most b variables of A such that the other atoms of the body of γ only use variables of X.

The formal definition of *suitable GTGDs* is then:

Definition 7.6. Letting w be the maximal width of GTGDs of Σ, a Σ-*suitable GTGD* is a GTGD which is full, is Σ-compatible, has exactly one principal guard, has breadth at most w, and has width at most w. When Σ is clear from context, we refer simply to a *suitable GTGD*.

Example 7.7. Let $n \in \mathbb{N}$ be an integer, let the signature consist of a principal relation R of arity n and of a single binary relation S for the side signature. For $w = 2$, the following full GTGD has width $\leq w$ but is not suitable (because it does not have breadth $\leq w$):

$$R(x_1 \dots x_n), S(x_1, x_2), \dots, S(x_{n-2}, x_{n-1}) \to S(x_{n-1}, x_n)$$

Note that the GTGDs of Σ, even the full GTGDs of Σ, may not all be suitable because they do not satisfy the breadth bound. Intuitively, the non-suitable full GTGDs of Σ will still be considered in the saturation process, but the process will only create new full GTGDs that are suitable.

We compute a bound on the number of suitable full GTGDs, as a function of the arity and size of the signature and of the side signature, together with the width and the size of the set Σ of GTGDs. This uses the fact that the suitable full GTGDs are Σ-compatible, have bounded width, and have bounded breadth:

Lemma 7.8. *The number of suitable full GTGDs is at most:*

$$|\Sigma|^2 \times (a+1)^{3w} \times 2^{n' \times w^{a'}}$$

where:

- $|\Sigma|$ *is the number of GTGDs in* Σ,
- a *is the maximal arity of any relation in* $\mathcal{S}$,
- n' *is the number of relations in the side signature* $\mathcal{S}'$,
- a' *is the maximal arity of the relations of* $\mathcal{S}'$,
- w *is the maximal width of a GTGD of* Σ.

Proof. We construct a suitable full GTGD by:

- Picking a principal guard atom A which is isomorphic to a head atom of Σ: this gives $|\Sigma|$ choices, and the resulting atom has at most a variables.
- Picking a subset of variables of A on which to add side facts: by the breadth bound we pick at most w of the a variables, so the number of choices can be overapproximated as $(a+1)^w$.
- Picking an instance of side facts on a domain of size at most w:
  - Each possible fact is obtained by picking a relation (among n'), and filling every position (of which there are at most a') with an element (of which there are at most w), i.e., there are at most $n' \times w^{a'}$ possible facts.
  - So, for the choice of sets of side facts, we have $2^{n' \times w^{a'}}$ options.
- Picking a head atom H which is isomorphic to a head atom of Σ: this gives at most $|\Sigma|$ choices, and again the head atom has at most a variables.
- Picking a sequence of exported variables from the body atom: this can be overapproximated as $(a+1)^w$ possible sequences.
- Picking a sequence of exported variables from the head atom: again $(a+1)^w$ possible sequences. The other variables are existentially quantified.

Putting it together, we obtain the claimed bound. □

Observe that, when w, n', and a' are all constant, then the above quantity is polynomial in the size of the input signature $\mathcal{S}$. Further, when only a' is bounded, then the quantity is singly exponential in the input.

Our goal in focusing on suitable full GTGDs is to identify which ones are *derived*, i.e., follow from Σ. We say that a suitable full GTGD γ is a *derived suitable full GTGD* if we have $\Sigma \models \gamma$, that is, any instance that satisfies Σ also satisfies γ. Again, the derived suitable GTGDs generally do *not* include all the full GTGDs of Σ, because not all of them are suitable.

For our proof of Theorem 7.3, we will need to show that the set of derived suitable full GTGDs can be computed efficiently. Let us define a specific kind of derived suitable full GTGD, namely, the *trivial* ones:

Definition 7.9. We say that a full GTGD γ is *trivial* if its head atom already occurs in its body. A *trivial suitable* full GTGD is a full GTGD which is both trivial and suitable.

We also define from Σ the set Σ_{triv} of trivial full GTGDs where the body contains a principal atom A which is an isomorphic copy of a GTGD head of Σ, the other atoms of the body are side atoms on at most w different variables of A, and the head is identical to A. The full GTGDs of Σ_{triv} are all trivial, they are all Σ-compatible, and they all satisfy the breadth bound: but they do generally not satisfy the width bound.

Note that not all trivial GTGDs are suitable, because even trivial GTGDs may have unbounded width and unbounded breadth. This is why we will restrict to derived suitable trivial GTGDs (to be put in our saturation), and the GTGDs of Σ_{triv} (which will be considered in the saturation process but will not be part of the saturation, because they are not suitable).

We now define the *saturation*, which is computed by starting with the suitable trivial full GTGDs and the suitable full GTGDs of Σ, and closing under the application of two rules, (Transitivity) and (Principal+Transitivity). Intuitively, (Transitivity) can be used to deduce new full GTGDs by combining full GTGDs with those already deduced; and (Principal+Transitivity) has the same purpose but where we additionally perform a rewriting of one of the GTGDs via a principal GTGD. (Recall from Definition 3.3 that a principal GTGD is a GTGD whose head is a principal relation.)

Definition 7.10. Given a set of GTGDs Σ that strongly obey side signature $\mathcal{S}'$, the $\mathcal{S}'$-*suitable saturation* $\widehat{\Sigma}$ is obtained by starting with the suitable full GTGDs in Σ, plus the trivial suitable full GTGDs, and applying the following inference rules until we reach a fixpoint:

- (Transitivity): Suppose that $\widehat{\Sigma} \cup \Sigma_{\text{triv}}$ contains n full GTGDs with the same body (up to renaming), that is, it contains full GTGDs $\beta \to B_1(\vec{z}_1), \ldots, \beta \to B_n(\vec{z}_n)$.

  Suppose that there is a full GTGD $\beta' \to \rho'$ in $\Sigma \cup \widehat{\Sigma}$, and that there is a homomorphism υ mapping β' to $\beta \wedge \bigwedge_j B_j(\vec{z}_j)$. Then add to $\widehat{\Sigma}$ the following if it is suitable:

$$\beta \to \upsilon(\rho').$$

- (Principal+Transitivity) Suppose that $\widehat{\Sigma} \cup \Sigma_{\text{triv}}$ contains n full GTGDs with the same body (up to renaming), that is, it contains full GTGDs $\beta \to B_1(\vec{z}_1), \ldots, \beta \to B_n(\vec{z}_n)$. Let δ_{cc} be a principal GTGD of Σ. We use the subscript cc to emphasize that the dependency "creates a child" in the chase. Let A_{cc} be its principal guard, let β_{cc} be the conjunction of its side atoms, and let H_{cc} be its head. Let δ' be a full GTGD of $\widehat{\Sigma}$ whose principal

guard A' is isomorphic to H_{cc}: up to renaming the variables of δ' we assume that A' is identical to H_{cc}. Let β' be the conjunction of the side atoms of δ' and let H' be its head atom. Assume that β' and H' only use variables of H_{cc} that are exported variables of δ_{cc}. Further assume that $A_{cc} \wedge \beta_{cc} \wedge \beta'$ can be mapped by a homomorphism v to $\beta \wedge \bigwedge_j B_j(\vec{z}_j)$. Then add the following full GTGD to $\widehat{\Sigma}$ if it is suitable:

$$\beta \to v(H')$$

Observe that (Principal+Transitivity) is quite similar to (Transitivity), but intuitively we are additionally composing with a principal GTGD δ_{cc} of Σ. We also note that a similar saturation process in discussed in [AB22], in the specific case of accessibility axioms for access methods, namely in the proof of [AB22, Proposition 6.6]. Other algorithms for "Datalog rewriting" of GTGDs have used similar closure rules – e.g. the FullDR algorithm of [BBG+24]. As in those cases, one has a closure rule that composes full GTGDs –in our case (Transitivity). And one also has a closure rule – in this case, (Principal+Transitivity) – that composes full GTGDs and another GTGD that may be non-full, provided that one can compose in a way that gives a full GTGD.

One rough intuition for these rules comes from arguing inductively that full GTGDs should suffice to capture the generation of facts on a given node v in a principal-exempt one-pass chase. One way that a fact F can come into node v is that we create a child c of v in the chase, using a principal GTGD δ_{cc} in the original set Σ, then generate a fact F in c, and propagate F rootwards back to v. We can break up the chase into three parts: (1.) generating the facts in v required to fire δ_{cc} and the other facts which will be inherited in c and used to generate F; (2.) the firing of δ_{cc}; and (3.) a chase sequence on c that generates F. Step (1.) will intuitively be inductively captured by the derived dependencies $\beta \to B_1(\vec{z}_1), \ldots, \beta \to B_n(\vec{z}_n)$; step (2.) corresponds to δ_{cc}; and step (3.) corresponds to the full GTGD δ'. The (Principal+Transitivity) inference rule is used to create a full GTGD that captures the composition of the whole process and can be applied directly on node v. We point the reader to Case 2 in the proof of Claim 7.13 within the completeness argument, where this is explained more formally.

Example 7.11. We first illustrate the inference rule (Transitivity). Assume that $\Sigma \cup \widehat{\Sigma}$ contains the following full GTGDs:

$$\begin{aligned} R(x, y_1, \ldots, y_n, z), S(x) &\to T(y_1) \\ &\vdots \\ R(x, y_1, \ldots, y_n, z), S(x) &\to T(y_n) \\ R(x, y_1, \ldots, y_n, z), T(y_1), \ldots, T(y_n) &\to U(z) \end{aligned}$$

Note that these GTGDs obey the side signature $\{S, T, U\}$ and have width at most 1; but the last one does not have bounded breadth. The (Transitivity) inference rule allows us to deduce the following full GTGD, which has width 1 and breadth 1:

$$R(x, y_1, \ldots, y_n, z), S(x) \to U(z)$$

We now illustrate the inference rule (Principal+Transitivity). Assume that the principal signature contains a 4-ary relation R and a ternary relation R', and that all other relations

are in the side signature. Assume that $\widehat{\Sigma}$ contains the following full GTGDs:

$$\gamma_1 : R(x_1, x_2, y_1, y_2), S(x_1), S(x_2) \to T(y_1)$$
$$\gamma_2 : R(x_1, x_2, y_1, y_2), S(x_1), S(x_2) \to T(y_2)$$

And it also contains the full GTGD

$$\gamma_3 : R'(y_1, y_2, z), T(y_1), T(y_2) \to U(y_1, y_2)$$

Note that these GTGDs have breadth 2 and width at most 2. Assume that Σ contains the following non-full GTGD of width 2 and breadth 2:

$$\gamma_{cc} : R(x_1, x_2, y_1, y_2), S(x_1), S(x_2) \to \exists z' R'(y_1, y_2, z)$$

Applying the inference rule (Principal+Transitivity), we deduce:

$$R(x_1, x_2, y_1, y_2), S(x_1), S(x_2) \to U(y_1, y_2)$$

This captures the effect of applying GTGDs γ_1 and γ_2 to get the additional T-facts, then the non-full GTGD γ_{cc} to get the R'-fact (which guards the T-facts), and finally the GTGD γ_3.

To understand why the inference rule is written the way it is, note that a naïve composition of γ_3 via γ_{cc} would have given:

$$R(x_1, x_2, y_1, y_2), S(x_1), S(x_2), T(y_1), T(y_2) \to U(y_1, y_2)$$

And the inference of such a rule would indeed have been generated in a saturation algorithm that was not concerned with the size. But this rule would have breadth 4. For this reason, the (Principal+Transitivity) inference rule intuitively performs a step analogous to (Transitivity) but via a principal GTGD, all in one go.

Further note that the inference rule (Principal+Transitivity) also applies for principal full GTGDs of Σ. For example, consider the modification of the last case above, where R' is a binary relation without its last position. Then γ is a full GTGD, and (Principal+Transitivity) can be applied in the same way.

We finish by exemplifying the purpose of considering the rules of Σ_{triv}, in the case of (Principal+Transitivity). Consider the following GTGDs, where the first is non-full and the second is full:

$$\gamma'_1 : R(x_1 \dots x_n), U_1(x_n) \to \exists x_{n+1} \cdots x_{2n} S(x_n \dots x_{2n})$$
$$\gamma'_2 : S(x_n \dots x_{2n}), U_2(x_n) \to U_3(x_n)$$

where U_1 and U_2 and U_3 are unary side relations and the other relations are principal. The following is a derived suitable full GTGD:

$$\gamma'_3 : R(x_1 \dots x_n), U_1(x_n), U_2(x_n) \to U_3(x_n)$$

To derive it and add it to $\widehat{\Sigma}$, we use (Principal+Transitivity) with the following trivial full GTGD of Σ_{triv}:

$$\gamma' : R(x_1 \dots x_n), U_1(x_n), U_2(x_n) \to R(x_1 \dots x_n)$$

This GTGD is not suitable, because it does not satisfy the width bound; but it is needed to give us a way to have $\beta = \{R(x_1 \dots x_n), U_1(x_n), U_2(x_n)\}$.

Clearly, by definition, all the GTGDs of $\widehat{\Sigma}$ are suitable. We now state and prove that the computation of the saturation can be performed efficiently:

Lemma 7.12. *There is a polynomial P such that, for any set Σ of GTGDs of width at most w which strongly obey side signature $\mathcal{S}'$, letting a be the arity of the signature, n' the number of relations in the side signature, and a' the arity of the side signature, we can compute $\widehat{\Sigma}$ in time $P(|\Sigma|, a^{O(w)}, 2^{n' \times w^{a'}})$.*

Proof. From our bound in Lemma 7.8, and knowing that the full GTGDs in the saturation are suitable, we know that the maximal size of $\widehat{\Sigma}$ satisfies our running time bound. We also know by an immediate variant of Lemma 7.8 that the number of GTGDs in Σ_{triv} satisfies the running time bound, because GTGDs of Σ_{triv} can be obtained by picking the body of a suitable GTGD, and then picking a head which is identical to the guard atom. So in the algorithm below, we materialize Σ_{triv}.

We can compute $\widehat{\Sigma}$ by iterating the possible production of rules until we reach a fixpoint, so it suffices to show that at each intermediate state of $\widehat{\Sigma}$, testing every possible inference rule application is in PTIME in $|\widehat{\Sigma} \cup \Sigma|$ for $\widehat{\Sigma}$ the set of derived suitable full GTGDs that have currently been computed.

We first explain how to test for applications of the (Transitivity) inference rule. We first make the bodies of the rules of $\widehat{\Sigma}$ canonical by first giving canonical names to the variables of the principal guard atom, in the order in which they appear, and then sorting the side signature atoms in some canonical way (e.g., by relation, then by lexicographic order on the variables). This can be done in polynomial time, and makes it easy to regroup all the GTGDs of $\widehat{\Sigma}$ that have an isomorphic body, and then try each possible body as a choice of β.

Now, for each choice of body β, we can consider all GTGDs $\beta \to B_1(\vec{z}_1), \ldots, \beta \to B_n(\vec{z}_n)$ of $\widehat{\Sigma}$ having body β, and consider the union H of their heads. Indeed, note that when applying (Transitivity) we can always assume without loss of generality that we consider all full GTGDs of $\widehat{\Sigma}$ having the body β, because having more such GTGDs will give us more rule heads $B_j(\vec{z}_j)$, which makes it easier to apply (Transitivity) to some choice of GTGD $\beta' \to \rho$. Now, we enumerate all GTGDs in $\Sigma \cup \widehat{\Sigma}$ and we test whether their body β' can be mapped homomorphically to $\beta \cup H$, and apply the inference rule (Transitivity) if that is the case. We must argue that this test can be done in PTIME. We do this by considering each principal atom A in $\beta \cup H$, testing in PTIME for each of them whether the principal guard of β' can be mapped homomorphically to A, and see whether the mapping thus defined is a homomorphism from β' to $\beta \cup H$. If this is the case, we can build the new full GTGD and add it to $\widehat{\Sigma}$ if it is suitable (which is easy to verify).

We now explain how to test for possible applications of the (Principal+Transitivity) inference rule. We make the GTGD bodies of $\widehat{\Sigma} \cup \Sigma_{\text{triv}}$ canonical like in the previous case: consider every GTGD body β in $\widehat{\Sigma} \cup \Sigma_{\text{triv}}$, and for each β consider all GTGDs of $\widehat{\Sigma}$ having β as body. Now, we consider every choice of a principal GTGD δ_{cc} of Σ and a full GTGD δ' of $\widehat{\Sigma}$. We test if these satisfy the conditions to apply the inference rule (Principal+Transitivity). Reusing the notations used in the inference rule, we check that the principal guard A' of γ' is isomorphic to H_{cc}: if it is not then we cannot apply the inference rule for this choice of δ_{cc} and δ'; if is is isomorphic then this defines the isomorphism ι and we continue with trying to apply the inference rule. We now define β'' and H'' and check that $A_{cc} \wedge \beta_2 \wedge \beta''$ maps homomorphically to $\beta \wedge \bigwedge_j B_j(\vec{z}_j)$ by defining first the homomorphism from A_{cc} to the principal atom of β, and checking if this mapping is a homomorphism from $A_{cc} \wedge \beta_2 \wedge \beta''$ to $\beta \wedge \bigwedge_j B_j(\vec{z}_j)$. If this check succeeds, we build the new full GTGD and add it to $\widehat{\Sigma}$ if it

is suitable. As noted earlier, suitability is easy to verify. Again, this can all be performed in PTIME, since we are dealing with sets of ground atoms guarded by a single (unique) principal ground atom, and thus there is only one possible isomorphism.

Hence, we can compute the process by testing every possible inference rule application on the current $\widehat{\Sigma}$, building the possible new GTGDs, and adding them to $\widehat{\Sigma}$ if they are new and suitable. This can be done in polynomial time in the dependencies Σ and in the current size of the set $\widehat{\Sigma}$. We can continue this process as long as the size of $\widehat{\Sigma}$ increases. From the bound on the maximal size of $\widehat{\Sigma}$, we conclude that the running time bound is respected. □

Let us now conclude the proof of Theorem 7.3 by showing that the full GTGDs of Σ, together with $\widehat{\Sigma}$, are actually a childish saturation in the sense of Definition 7.2. Towards showing this, we first establish that $\widehat{\Sigma}$ contains all derived suitable full GTGDs:

Claim 7.13. The set of GTGDs $\widehat{\Sigma}$ is exactly the set of derived suitable full GTGDs.

Proof. It is clear by definition that the full GTGDs in $\widehat{\Sigma}$ are all suitable, and an immediate induction shows that they are all derived (i.e., they all logically follow from Σ). So let us focus on the converse: let us consider a derived suitable full GTGD $\gamma : A \wedge \beta \to H$, and show that it is in $\widehat{\Sigma}$. From the width bound on $\gamma \in \Sigma$, since γ is full, we know that H has at most w different variables.

As γ is derived, let us pick any isomorphism ι to transform the principal guard A, the side atoms β, and the head H respectively into: a principal fact $A_0 := \iota(A)$, a set of side facts $\beta_0 := \iota(\beta)$, and a fact $H_0 := \iota(H)$. Let us consider a proof of H_0 from $I_0 = \{A_0\} \cup \beta_0$. Using Theorem 5.6 let us more precisely take a principal-exempt one-pass chase proof of H_0 from I_0 using the GTGDs of Σ. As Σ is homomorphism-closed, we can assume without loss of generality that, in all GTGD firings, no two distinct exported variables are mapped to the same domain element.

We show that γ is in $\widehat{\Sigma}$ by induction on the length of such a principal-exempt one-pass chase derivation of H_0 from I_0 using Σ. The base case corresponds to the case of an empty derivation, in which case $H_0 \in I_0$ so that γ is a trivial GTGD and we immediately conclude because all trivial suitable full GTGDs are in $\widehat{\Sigma}$ by definition.

To show the induction step, assume without loss of generality that the principal-exempt one-pass chase proof of H_0 from I_0 finishes by deriving H_0, and let v be the node on which the last chase step is performed to derive H_0. We distinguish two cases, depending on whether v is the root node v_0 of the tree-like chase sequence, or whether v is a strict descendant of v_0.

Case 1: the last firing is performed on the root v_0. If $v = v_0$, then the last chase step fired a GTGD γ' of Σ on a trigger τ. The GTGD γ' must be a full GTGD because it derived the fact H_0, which contained only values from v_0. In the trigger τ, the principal guard of γ' was mapped to the principal guard A_0 of v_0. We can assume by homomorphism-closure that the mapping is a bijection on the exported variables. If γ' has no side atoms, then we conclude immediately by applying (Transitivity) with the full trivial GTGD $A \wedge \beta \to A$ of Σ_{triv}, together with γ'. So in what follows we assume that γ' has some side atoms.

The side atoms of γ' were mapped to some facts β'_0 of v_0 which were either part of β_0 or were derived earlier by the chase sequence. In other words, for each fact F of β'_0, there is a principal-exempt one-pass chase proof of F from I_0 (possibly of length 0), which witnesses that the full GTGD $\gamma_F : A \wedge \beta \to \iota^{-1}(F)$ is a logical consequence of Σ. Now, we can argue that, for each F, the GTGD γ_F is suitable. It is obviously a full GTGD. In terms of the

compatibility requirements, it has the same body as the suitable full GTGD γ so it satisfies the breadth bound. It has exactly one principal guard, the principal guard is Σ-compatible, and its head is a side atom so it is also Σ-compatible. Finally, its head atom is for a side signature relation, so the width of γ_F is at most the arity of that atom, i.e., at most the side signature arity.

Further, γ_F is a derived suitable full GTGD, which has a strictly shorter principal-exempt one-pass chase proof from I_0 and Σ. Thus, by the induction hypothesis, $\gamma_F \in \widehat{\Sigma}$. Now, applying (Transitivity) to the full GTGDs γ_F of $\widehat{\Sigma}$ for the non-empty set $F \in \beta'_0$ and to the full GTGD γ' of Σ, we conclude that our initial full GTGD γ, which is suitable by hypothesis, was in $\widehat{\Sigma}$.

Case 2: the last firing is performed on a strict descendant of the root node v_0. If the last firing was performed on a node $v \neq v_0$, then v is a strict descendant of v_0. Let v_1 be the child of v_0 which is an ancestor of v; possibly $v = v_1$. Let A_1 be the principal guard of v_1, and let β_1 be the facts inherited from v_0 when creating v_1: by definition of the principal-exempt one-pass chase, the facts of β_1 are all side facts. Let γ_{cc} be the GTGD of Σ (full or non-full) which was fired earlier to create node v_1. By definition of the principal-exempt one-pass chase, γ_{cc} is a principal GTGD, and its head was instantiated to A_1. Let τ_{cc} be the trigger fired when creating v_1. We know that τ_{cc} consists of the principal guard A_0 of the root v_0 together with some side facts β'_0 which were present in v_0 at that moment. And thanks to homomorphism-closure we know the following fact (*): in τ_{cc} no two distinct exported variables are mapped to the same element. Let us rename the exported variables of γ_{cc} to match the variables of the guard A of γ, and let us rename the existentially quantified variables of γ_{cc} to use fresh variable names. We can extend the isomorphism ι, which was originally defined on the variables of A, and extend it to an isomorphism defined on the variables of A and also on the head of γ_{cc}, which is mapped to the fact A_1. We can then split the chase sequence into three successive parts:

- The initial part, which starts with I_0 and creates the facts of β'_0 in v_0 in some order. For each fact $F \in \beta'_0 \setminus \beta_0$, we now reason as in Case 1 above to show that the full GTGD $\gamma_F : A \wedge \beta \to \iota^{-1}(\beta'_0)$ is suitable and is in $\widehat{\Sigma}$: the latter uses the induction hypothesis.
- The firing of τ_{cc} for the principal GTGD γ_{cc} of Σ, which creates v_1 containing the head instantiation A_1 along with some facts $\beta_1 \subseteq \beta'_0$ inherited from v_0.
- The subsequent part of the chase sequence, which is performed in the subtree rooted at v_1 by definition of the one-pass chase, and which derives the fact H_0 from $I_1 = \{A_1\} \cup \beta_1$. Note that H_0 must be on elements shared between A_0 and A_1

Now, let us consider the full GTGD $\gamma'' : \iota^{-1}(A_1) \wedge \iota^{-1}(\beta_1) \to H$. The third bullet point above witnesses that this full GTGD is a logical consequence of Σ. Let us show that it is suitable. It has precisely one principal guard. It obeys the width bound because H is the head of γ which is suitable by assumption so it has at most w different variables. It obeys the breadth bound because β_1 uses at most w different elements of A_1 thanks to the width bound on γ_{cc}. Now let us verify the compatibility conditions. The principal guard is Σ-compatible because A_1 was created by instantiating the head of γ_{cc} (without identifying any variables thanks to (*)), so $\iota^{-1}(A_1)$ is actually exactly the head of γ_{cc}. And the head H is the head of γ, and since γ is suitable its head is Σ-compatible. Thus, γ'' is a suitable derived full GTGD with a strictly shorter chase proof. By induction hypothesis, $\gamma'' \in \widehat{\Sigma}$.

We now distinguish two subcases: either $\beta'_0 \setminus \beta_0$ is empty, or it is non-empty. We first deal with the subcase where it is empty. Then we want to apply the inference rule

(Principal+Transitivity) with $\gamma_{cc} \in \Sigma$, $\gamma'' \in \widehat{\Sigma}$, and the trivial full GTGD of Σ_{triv} with same body as γ, namely: $A \wedge \beta \to A$. Let us show that the inference rule is indeed applicable. We know that the principal guard of γ'' is exactly the head of γ_{cc}. The facts of β_1 were inherited from v_0, and since $\beta'_0 \setminus \beta_0$ is empty they were all part of β_0. So the side atoms $\iota^{-1}(\beta_1)$ of γ'', together with the body of γ_{cc}, can be homomorphically mapped to $A \wedge \beta$, as required to apply the rule.

In the subcase where $\beta'_0 \setminus \beta_0$ is non-empty, we want to apply the inference rule (Principal+Transitivity) again with $\gamma_{cc} \in \Sigma$ and $\gamma'' \in \widehat{\Sigma}$, but this time together with the non-empty set of the γ_F in $\widehat{\Sigma}$ for $F \in \beta'_0 \setminus \beta_0$ instead of using a trivial full GTGD of Σ_{triv}. Let us show that the inference rule is indeed applicable. We already know that the principal guard $\iota^{-1}(A_1)$ of γ'' is exactly the head of γ_{cc}. The facts of β_1 were inherited from v_0, and we have explained that they were part of β_0 or were created by instantiating the heads of the GTGDs γ_F. Further, the body of γ_{cc} can be mapped to β'_0, so these facts were also part of β_0 or were created by instantiating the heads of the GTGDs γ_F. Thus, the side atoms $\iota^{-1}(\beta_1)$ of γ'', together with the body of γ_{cc}, can be homomorphically mapped to the conjunction of $A \wedge \beta$ and of the heads of the GTGDs γ_F, as required to apply the rule.

The application of (Principal+Transitivity) deduces $A \wedge \beta \to H$, namely our initial full GTGD γ, which is suitable by hypothesis. So γ is in $\widehat{\Sigma}$ also in the second case. This concludes the proof. □

We can now conclude the proof of Theorem 7.3, which will be direct from Claim 7.13:

Proof of Theorem 7.3. The running time bound was shown in Lemma 7.12, so we must only show that $\widehat{\Sigma}$ is a childish saturation as in Definition 7.10. We know from the easy direction of Claim 7.13 that all GTGDs in $\widehat{\Sigma}$ are logically entailed by Σ. In the rest of the proof, we show that $\widehat{\Sigma}$ is complete for fact entailment on childish instances.

Let I be a childish instance, let F be a fact on the domain of I such that $I, \Sigma \models F$. Let us show that $I, \widehat{\Sigma} \models F$. If $F \in I$, then there is nothing to show, so we assume that $F \notin I$. Now, considering a chase proof of $I, \Sigma \models F$, we know that F must be isomorphic to the head of a GTGD of Σ, namely, that of the last GTGD γ_H which is fired in a chase proof of F from I with Σ. Note that this uses homomorphism-closure of Σ, which implies that in rule firings, we can always assume that all exported variables of rules are mapped to distinct elements. We also know, from the width bound on Σ, that F uses at most w distinct elements of I, and since F is a fact on the domain of I it uses at most w distinct elements overall.

The childish instance I consists of a principal fact F' which is an isomorphic copy of the head of some GTGD γ_B from Σ, together with some side facts I' on at most w elements of F'. Let γ be the full GTGD obtained by renaming the elements of I and F from constants to variables, with I giving the body and F giving the head. We claim that γ is a suitable derived full GTGD. Indeed:

- γ has exactly one principal guard because I contains exactly one principal fact F' which contains all elements of $\mathsf{Adom}(I)$.
- γ is Σ-compatible, as witnessed by γ_H for the head atom and γ_B for the principal guard atom.
- γ has width at most w, because F uses at most w distinct elements.
- γ has breadth at most w because I' uses at most w elements of F'.

Thus, γ is a suitable full GTGD. Further, γ is a derived suitable full GTGD, because the chase proof of $I, \Sigma \models F$ witnesses that γ is logically entailed by Σ. Hence, by Claim 7.13, we immediately conclude that $\gamma \in \widehat{\Sigma}$. Hence, $I, \widehat{\Sigma} \models F$, which is what we wanted to show. □

8. Fact closure and making the chase similar to a linear chase

We are now ready for the second stage of our proof. For now, we have used an analysis of the principal-exempt one-pass tree-like chase to design an algorithm that computes a saturation of a finite set of GTGDs assuming it is given a childish instance.

We first show that we can use childish saturations to deal with the fact entailment problem on arbitrary instances.

Definition 8.1. Given an instance I and GTGDs Σ, we say that I is Σ-*fact-saturated* if it is closed under facts entailed on the same domain. That is, I is Σ-fact-saturated if for any fact F over $\mathsf{Adom}(I)$, if $I, \Sigma \models F$, then F is already in I.

Definition 8.2. If I is a subinstance of I' we say I' is I-*deactivated* if I' has no active triggers with image in I.

Proposition 8.3. *There is an algorithm that takes as input a set of GTGDs Σ that strongly obeys side signature $\mathcal{S}'$, along with an instance I, and performs chase steps with Σ and with a childish saturation $\widehat{\Sigma}$ of Σ to obtain an $I' \supseteq I$ that is Σ-fact-saturated and I-deactivated, in time* $\mathrm{Poly}(|I|^{O(w)}, |\Sigma|, a^{O(w)}, 2^{n' \times w^{a'}})$*, where a, w, n', a' are as in Lemma 7.12.*

Proof. We can compute a childish saturation $\widehat{\Sigma}$ in the required time, by Theorem 7.3. We then perform a variant of the one-pass chase of $\Sigma \cup \widehat{\Sigma}$ over I, but truncated to only one level, i.e., to the root node (initially a single node containing the facts of I) and to the immediate children of the root node. More precisely, the process repeatedly applies each of the following kinds of operations until saturation, starting with a singleton tree made of a root node v_0 containing the instance I:

- Firing all full rules of Σ and all rules of $\widehat{\Sigma}$ on v_0 to create new side facts on $\mathsf{Adom}(I)$, which are added to v_0 (we do this whenever such new facts can be added to v_0).
- Firing all non-full rules of Σ on every active trigger which is included in $\mathsf{Adom}(I)$, creating child nodes with a principal fact which is the head of the rule. We do this whenever such an active trigger exists, and create a new child v containing the head fact (a principal fact) and containing as inherited facts all side facts of v_0 that are guarded by the principal fact. Further, we fire all full GTGDs of $\widehat{\Sigma}$ on v. After this the trigger is no longer active.
- For triggers in I of principal GTGDs we create a new child for them, saturate using $\widehat{\Sigma}$, and propagate new $\mathsf{Adom}(I)$ facts back to the root v_0. More precisely, we do the following whenever it can add a new fact to v_0:
  - Fire a trigger for a principal GTGD of Σ – i.e., a rule with a principal atom in the head, full or non-full – to create a new child node v of v_0 containing one principal fact F and possibly some inherited side facts.
  - Fire all full GTGDs of $\widehat{\Sigma}$ on v to create new facts on $\mathsf{Adom}(F)$, which are added to v. Note that we deviate from the principal-exempt chase here in that we may create new principal facts on $\mathsf{Adom}(F)$, in which case they are added to the current node, i.e., they do not trigger the creation of a child node with a relaxed chase step.

– Propagate all facts in v from $\mathsf{Adom}(F) \cap \mathsf{Adom}(I)$, rootwards to v_0.
Note that we deviate again from the principal-exempt chase here because we are allowed to propagate principal facts rootwards – but because we never inherit principal facts, they will remain in v_0 and will not be inherited.

The final instance I' that we take is the set of the facts in v_0 at the end of this process.

We first claim that this process finishes in the required time bound. In the first top-level bullet, the number of possible facts to add to v_0 is upper bounded by $|\mathsf{Adom}(I)|^{O(a')}$ for a' the side signature arity, because we only create side facts in this bullet item. As $w \geq a'$, this is upper bounded by $|\mathsf{Adom}(I)|^{O(w)}$. For the second top-level bullet, the number of nodes we create is bounded by the number of possible head instantiations of non-full GTGDs of Σ, which can be bounded by choosing the non-full GTGD and the choice of exported elements, i.e., by $|\Sigma| \times |\mathsf{Adom}(I)|^{O(w)}$. Further, on each created node we fire all full GTGDs of $\widehat{\Sigma}$ which can create at most $|\widehat{\Sigma}| \times a^{O(w)}$ new facts in each node. For the third top-level bullet, the number of possible facts to create in I is upper bounded by $|\widehat{\Sigma}|$ (the number of rule heads) times again $|\mathsf{Adom}(I)|^{O(w)}$ (the number of instantiations of the exported variables of the firing that creates the fact), so the number of nodes that we create is upper bounded by $|\widehat{\Sigma}| \times |\mathsf{Adom}(I)|^{O(w)}$. For each created node we fire all full GTGDs of $\widehat{\Sigma}$, for which the bound is the same as for the previous bullet item.

So the number of chase steps performed satisfies the bound. Further, going over the GTGDs of Σ and $\widehat{\Sigma}$ and testing possible applications is in polynomial time in their size and in the current number of facts created, because we can test applicability by mapping the principal guards of GTGDs to each choice of principal fact, and then checking if this defines a homomorphism from the GTGD body, and if so add the head if it is a new fact. Remembering that the size of $\widehat{\Sigma}$ is bounded, following the running time bound of Lemma 7.12, we conclude that the running time bound is as required.

It is clear that the process only uses chase steps from $\Sigma \cup \widehat{\Sigma}$.

We next argue that I' is fact-saturated, which will rely on bullet items one and three, and we also will use the completeness of $\widehat{\Sigma}$ on childish instances. Assume by contradiction that I' is not fact-saturated. That is, there is a fact F on $\mathsf{Adom}(I')$ which is not in I' but is entailed by I' and Σ. There are two cases: either F contains an element of $\mathsf{Adom}(I') \setminus \mathsf{Adom}(I)$, or it does not. In the first case, the image of the trigger in I' must be in one of the child nodes of v_0 that we created; but these nodes contained a childish instance when they were created, and we have applied all GTGDs of $\widehat{\Sigma}$ to them, so because $\widehat{\Sigma}$ is complete on childish instances we must have also derived F in I', which is a contradiction. Hence, let us focus on the second case, when F is a fact over $\mathsf{Adom}(I)$.

In this case, consider a principal-exempt one-pass chase proof of F from I. We can assume without loss of generality that F is a minimal counterexample, in the sense that, in this proof, it is the first fact on $\mathsf{Adom}(I)$ which is derived in the proof but not present in I'. The GTGD firing that created F cannot have been applied to the root node v'_0 of the principal-exempt one-pass chase, otherwise all hypotheses to the firing are facts on $\mathsf{Adom}(I)$ which were present in I' by minimality, and so they were present in v_0, so that F should have been derived by the first bullet item above. Hence, the GTGD firing that created F must have applied in a strict descendant of v'_0 in the principal-exempt one-pass chase. This node, call it b, is a descendant of a child node b' of v'_0, and b' in turn was created by firing a principal GTGD γ of Σ on I. The assumption that γ is principal is because, by Proposition 4.4, non-principal GTGDs are full and so firing them does not create new nodes

in the principal-exempt chase. By minimality, all hypotheses to fire γ were present in I' as well, so the same firing could have been performed in our truncated variant of the chase. And by minimality we would have created a node inheriting the same facts from the root nodes, i.e., with the same set of facts I'' that were present in b' when it was created in the chase proof of F from I.

We now use the fact that I'' is a childish instance, because γ is a principal GTGD of Σ of width at most w. Thus, I'' consists of one principal fact which is an isomorphic copy of a GTGD head of Σ, together with side facts on at most w elements, corresponding to the exported variables of γ. Thus, the completeness of $\widehat{\Sigma}$ on the childish instance I'' ensures that the fact F, which is a fact on $\mathsf{Adom}(I'')$, was also derived by applying the rules of $\widehat{\Sigma}$ in our chase variant. This establishes a contradiction and concludes the proof of the second case of our case analysis. Hence, I' is fact-saturated.

We last argue that I' is I-deactivated. Assume by way of contradiction that I' contains an active trigger for a GTGD γ with image in I. If γ is full, then this contradicts fact-saturation which we established above. If γ is non-full, then this trigger should have been fired in the second bullet point, which is also a contradiction. Hence, I' is I-deactivated, which concludes the proof. □

We note a corollary that may be of independent interest:

Corollary 8.4. *For any fixed side signature $\mathcal{S}$ and width w, there is a polynomial time algorithm that takes as input a set of GTGDs Σ that strongly obeys side signature $\mathcal{S}'$ and have width at most w, along with an instance I, and computes a Σ-fact-saturated instance $I' \supseteq I$.*

In particular, the fact entailment problem is in polynomial time for such GTGDs.

The corollary implies in particular that fact entailment for linear TGDs of bounded width is in polynomial time, a result that does not seem to have been noted explicitly in the literature. Indeed, linear TGDs always vacuously strongly obey the empty side signature (up to performing homomorphism-closure, which is in 2^w hence constant for fixed w).

If we were interested only in atomic queries – single atoms with no quantifiers – instead of OWQA with general CQs, then Proposition 8.3 would suffice to conclude our main results. Note that for general CQ query answering it is also possible to reduce the query answering process to applying a set of full TGDs: general CQs are *Datalog-rewritable* with respect to GTGDs, see [BBtC18]. But we do not see a way to use these methods to take advantage of the side signature and get the refined complexity bounds we need.

Instead we will continue to use tree-like chase proofs to do query answering, but look at the impact of adding all of these derived GTGDs on the chase process. Recall that in the tree-like chase we have chase steps and propagation steps. Our goal is to get a chase with no propagation steps at all, as discussed in the proof overview of Section 6. To do that we will add *shortcuts* which summarize the impact of chase steps combined with propagation steps. We will do so by defining a tree-like chase called the *shortcut chase*. Note that this chase is tree-like but it is *not* a principal-exempt chase; it will not feature relaxed chase steps, and will not create new nodes when firing full principal GTGDs. The principal-exempt chase was only used in the previous section and in Proposition 8.3; we do not use it here and we will not use it in the sequel.

Let Σ be the set of GTGDs of width w that strongly obeys side signature $\mathcal{S}'$, and let Σ' be a childish saturation of Σ. (We will of course take Σ' to be the closure $\widehat{\Sigma}$ defined in the previous section.) The *shortcut chase* based on Σ' is a tree-like chase sequence of the form

described below. Informally we apply the set of non-full GTGDs of Σ and the full GTGDs of Σ' in alternation.

A shortcut chase starts with a chase tree consisting of a single root node T_0, then consists of two alternating kinds of steps:

- The *non-full steps*, where we fire a non-full GTGD of Σ on a node g. The chase step creates a new node g' which is a child of g, which contains the result F' of firing the non-full GTGD along with a copy of the side facts of g which only use elements shared between F and F' (i.e., all side facts that can be inherited are inherited).
- The *full saturation steps*, which apply to a node g, only once per node, precisely at the moment where it is created by a non-full step. In this step, we apply all the full GTGDs of Σ' to the facts of g, and add the consequences to g (they are still on the domain of g because the rules are full).

Note the absence of propagation steps: the facts generated in a node are never propagated rootwards. Also note that whenever we create a node then the facts that it contains are a childish instance.

Our next goal is to argue that the shortcut chase is as good as the usual chase in terms of query answering. As we are working with childish saturations, which are only complete for entailment on childish instances, we will use Proposition 8.3 to ensure that the original instance is fact-saturated.

Proposition 8.5. *For any childish saturation Σ' of Σ, the shortcut chase based on Σ' emulates Σ on any fact-saturated instance: for each fact-saturated instance I and each Boolean CQ Q, we have $I, \Sigma \models Q$ if and only if Q holds in an instance produced from I by a shortcut chase sequence with the full GTGDs of Σ' and the non-full GTGDs of Σ starting at I.*

Proof. The direction from right to left is simple, and does not use fact-saturation of I. Shortcut chase steps are chase steps, and the dependencies in the childish saturation Σ' are entailed by Σ. Thus, it is clear that if Q holds in an instance produced from I using shortcut chase steps, then we have a chase proof witnessing that $I, \Sigma \models Q$.

We prove the other direction, assuming $I, \Sigma \models Q$. We first show the following auxiliary claim (*): in a shortcut chase that starts with a fact-saturated instance, when we derive a fact F in a node v (which by fact-saturation is not the root node), then F is not guarded in the parent of v. Indeed, assume by contradiction that F is guarded in a node v' of F but was derived in a child v of v'. The set of facts present in the node v' when it was created formed a childish instance I', and the shortcut chase sequence witnesses that the fact F on $\mathsf{Adom}(I')$ is entailed by I' and Σ', hence by I' and Σ. So by completeness of the childish saturation Σ', we should have derived F in the full saturation step on v and inherited it in v' instead of deriving it on v' as we assumed — a contradiction.

We next show that (*) implies the following claim (**): in a shortcut chase that starts with a fact-saturated instance, every side fact occurs in every node in which it is guarded. To show this, first notice that side facts which are in the initial instance will be inherited as long as they are guarded, so there is nothing to show for them. As for side facts which are not in the initial instance, they are always created by full saturation steps, because the GTGDs of Σ strongly obey the side signature, so non-full GTGDs are all principal. Let us consider a side fact F derived on a node v by a full saturation step. When this happens, v is a leaf node, and we claim that there is no other node v' that guards F. Indeed, if there were another node v' which does guard the fact, then let v'' be the least common ancestor of v

and v'. It must be the case that v'' also guards F, otherwise the values of $\mathsf{Adom}(F)$ cannot have been re-introduced in v and in v'' independently. So since v is a leaf node when F is created, we have $v \neq v''$. This implies that, considering the leafward path from v'' to v, the parent of v also guards $\mathsf{Adom}(F)$. But it does not contain F, and we have a contradiction of (*). Thus, when F is created in occurs in the only node in which it is currently guarded. Then, as F is a side fact, it will be inherited leafwards in all nodes where it is guarded. Thus, we have established (**).

We are now ready to show that the shortcut chase is complete, using (**). We show that for every tree-like chase proof $T_0, \ldots, T_m$ which starts from a fact-saturated instance I, there is a shortcut chase proof $T'_0, \ldots, T'_m$ from I with an isomorphism from the underlying instance of T_m to that of T'_m which is the identity on the active domain of I. Thus, in particular, if $T_0, \ldots, T_m$ witnesses that $I, \Sigma \models Q$, then Q holds in the instance underlying T_m, and taking the image of a match by ι_m we see that the same is true on T'_m.

Let us proceed by induction on the length of the tree-like chase proof $T_0, \ldots, T_m$. The base case is a tree-like chase proof of length 0, in which case we simply pick ι_0 to be the identity on I.

For the induction step, consider a tree-like chase proof $T_0, \ldots, T_m$ with $m > 0$. We immediately apply the induction hypothesis to obtain a shortcut chase proof $T'_0, \ldots, T'_{m'}$ such that there is an isomorphism ι_{m-1} from the underlying instance of T_{m-1} to that of $T_{m'}$ which is the identity on $\mathsf{Adom}(I)$, and let us explain how to continue the shortcut chase proof to obtain a new tree $T'_{m'+1}$ and an isomorphism ι_m from the underlying instance of T_m to that of $T'_{m'+1}$ which is the identity on $\mathsf{Adom}(I)$.

We first eliminate two easy cases. First, T_m may be obtained by applying a propagation step. Second, T_m may be obtained by performing a chase step in a node v which derives a fact that already occurs somewhere in T_{m-1}. In both these cases, we can conclude immediately using $T'_0, \ldots, T'_{m'}$ and ι_{m-1} as a witness. So we assume that T_m is obtained by applying a chase step, and let F be the fact produced. We distinguish two cases depending on whether the GTGD γ used in the chase step is full or non-full.

First, if the GTGD γ is full, consider the image of its trigger in T_{m-1}, calling it S. Because γ is a rule of Σ which strongly obeys the side signature, we know that S is formed of a principal fact F_0 and side facts β_0 on $\mathsf{Adom}(F_0)$. Given that F is a new fact, as the instance I is fact-saturated, we know that $\mathsf{Adom}(F) \not\subseteq \mathsf{Adom}(I)$, so since γ is full we know that $\mathsf{Adom}(F_0) \not\subseteq \mathsf{Adom}(I)$. So letting $F'_0 := \iota_{m-1}(F_0)$, we know that F'_0 occurs in a node n' of $T'_{m'}$ which is not the root. The facts of $\iota(\beta_0)$ also occur in $T'_{m'}$ and are guarded by n', so by (**) they also occur in n'. If n' already contains the fact $\iota_{m-1}(F)$ within $T'_{m'}$, then we can just conclude immediately with $T'_0, \ldots, T'_{m'}$ and ι_{m-1}. Otherwise, we can continue the shortcut chase proof by firing γ on n' which contains the trigger $\iota_{m-1}(S)$, and deduce the fact $\iota_{m-1}(F)$ in n'. This gives us $T'_{m'+1}$ which admits an isomorphism $\iota_m := \iota_{m-1}$ from $T_{m'}$ to $T'_{m'+1}$ which is the identity on $\mathsf{Adom}(I)$.

We now consider the case where the GTGD γ is non-full. Consider again the image S of its trigger in T_{m-1}. We want to find a node n' of $T'_{m'}$ on which the same firing can be applied. Again S is formed of a principal fact F_0 and side facts β_0 on $\mathsf{Adom}(F_0)$. This time it may be the case that F_0 is a fact of I. In this subcase β_0 are facts over $\mathsf{Adom}(I)$ which are entailed by I so $\beta_0 \subseteq I$ because I is fact-saturated. So in this subcase we can take n' to be the root node of $T'_{m'}$, which contains the facts of $\iota_{m-1}(S)$. In the subcase where F_0 is not a fact of I, then we reason as in the previous case: $\iota_{m-1}(F_0)$ occurs in a non-root node

of $T'_{m'}$, the facts of $\iota_{m-1}(\beta_0)$ are guarded in that node, so also occur there by (**). In this subcase we define n' to be a node of $T'_{m'}$ that contains the facts of $\iota_{m-1}(S)$.

In both these subcases we can perform a non-full step and apply γ with trigger mapping to $\iota_{m-1}(S)$, within node n' to create the tree $T'_{m'+1}$ with a child n'' of n' and a fact F' isomorphic to F. We can extend ι_{m-1} to ι_m by mapping the new values introduced in the chase step from T_{m-1} to T_m to the isomorphic values in F'. This gives ι_m, which is an isomorphism from the underlying instance of T_m to that of $T'_{m'+1}$, and is the identity on $\mathsf{Adom}(I)$ as required.

So, both for full GTGDs and non-full GTGDs, we have successfully extended the tree-like chase sequence while preserving an isomorphism which is the identity on I, so every query that admits a tree-like proof also admits a proof with the shortcut chase. This concludes the proof. □

We will explain in the next section how to translate the shortcut chase to a set of linear TGDs.

9. The final stage: Linearization and its justification

We now describe the third and last stage of the proof of Theorem 3.6. We describe the translation to a set of linear TGDs, intuitively introducing predicates for each guarded set of atoms on the side signature; and also describe the pre-processing of the instance, intuitively closing under the full dependencies that we introduce. The correctness of this transformation will then be argued using the shortcut chase studied in the previous section.

Recall the definition of childish instances (Definition 7.1). In the definition of the linearization, we will refer to *types*:

Definition 9.1. A *type* is an isomorphism type of a childish instance, i.e., an equivalence class of the childish instances quotiented under the isomorphism equivalence relation.

For each type θ, create a predicate R_θ whose arity is that of the principal fact of θ. Observe that this creates a singly exponential number of relations when the arity a' of $\mathcal{S}'$ is fixed, and it creates only polynomially many relations when we further fix the arity of the signature and also fix the full side signature $\mathcal{S}'$. We then let $\mathsf{Linearize}(\Sigma)$ consist of the linear TGDs defined as follows. For every type θ, fix a childish instance S on domain $\vec{x}$ achieving type θ, and write $P(\vec{x})$ for the principal fact of S and $\vec{y} \subseteq \vec{x}$ for the elements of $\vec{x}$ on which there are side facts (there are at most w). Also fix a homomorphism[1] h from $\vec{y}$ to itself, and extend h to a homomorphism from $\vec{x}$ to itself which is the identity on elements that do not occur in $\vec{y}$. Let S' be the instance obtained by starting with $h(S)$ and then repeatedly applying the full TGDs of $\widehat{\Sigma}$ (a full saturation step). Then our linearization will contain:

• (Instantiate): For every fact $A(h(\vec{x}))$ of S', the full linear TGD:

$$R_\theta(h(\vec{x})) \to A(h(\vec{x}))$$

Note that in particular we always have the TGD:

$$R_\theta(h(\vec{x})) \to P(h(\vec{x}))$$

[1]Note that we could avoid considering such homomorphisms if we required that the saturation is homomorphism-closed: this could be achieved in the required bounds just as in the case of the input constraints. However, in the present section we think it is simpler to consider the homomorphisms directly.

- (Lift): For every non-full GTGD δ in Σ of the form $\beta(\vec{z}) \to \exists \vec{w}\ T(\vec{z}, \vec{w})$ where $\beta(\vec{z})$ has a match h' in S', we add the linear TGD:

$$R_\theta(h(\vec{x})) \to \exists \vec{w}\ R_{\theta''}(h'(\vec{z}), \vec{w})$$

where θ'' is the type of the childish instance obtained as follows:
  - start with a principal fact $F = T(h'(\vec{z}), \vec{a})$ where $\vec{a}$ are nulls, intuitively corresponding to the result of firing δ;
  - add the side facts of S' that use only elements of $h'(\vec{z})$, intuitively corresponding to inherited facts.

Informally, the (Lift) rules simulate firing of non-full rules in the shortcut chase, and the (Instantiate) rules move us back into the original signature from the lifted signature.

Note that the full GTGDs are not explicitly mentioned in these linearized rules. However, they play a role in two places. First, within (Instantiate) and (Lift) rules, we are computing derived facts in computing S', and this involves the full GTGDs. Secondly, we will be running these rules on a pre-processed instance which is already fact-saturated.

We give a brief example to show how the transformation works:

Example 9.2. Let Σ consist of the ID $R(x, y) \to \exists z\ R(y, z)$ and the full GTGD $R(x, y), U(x) \to U(y)$.

- Our side signature here will consist of only U; thus the only principal relation is R;
- The maximal arity of a side signature atom, denoted a', is 1;
- The maximal width w is 1, which is $\geq a'$.

We have three childish instances up to isomorphism:

- $I_1 = \{R(1, 2), U(1)\}$, corresponding to type θ_1
- $I_2 = \{R(1, 2), U(2)\}$, corresponding to type θ_2
- $I_3 = \{R(1, 2)\}$, corresponding to type θ_3

In the linearization we thus have fresh relations R_{θ_1} and R_{θ_2} and R_{θ_3}.

The rule (Instantiate) will give us full linear TGDs such as:

$$R_{\theta_1}(x, y) \to U(x)$$
$$R_{\theta_1}(x, y) \to R(x, y)$$

The rule (Lift) will provide us with linear TGDs such as:

$$R_{\theta_2}(x, y) \to \exists z\ R_{\theta_1}(y, z).$$

We will show in the sequel that the resulting linear TGDs allow us to solve OWQA for the original dependencies, intuitively because they amount to performing the shortcut chase.

The result of our transformation clearly consists of linear TGDs. Further, they are of semi-width w: indeed, the rules produced by (Lift) have width bounded by w because the same is true of the non-full GTGDs of Σ, and the rules produced by (Instantiate) have an acyclic position graph.

We have now given the algorithm for computing Linearize(Σ). We now argue that these rules can be computed efficiently:

Claim 9.3. For any constant bound $a' \in \mathbb{N}$ on the arity of the side signature, there are fixed polynomials P_1 and P_2 such that the number of rules in Linearize(Σ) is in $P_1(|\Sigma| \times a \times w)^{P_2(w, n')}$ and the time to construct them is polynomial in their number.

The proof for this is similar to the argument for the bound shown in Lemma 7.8.

Proof. Let n be the number of relations of the signature, a the maximal arity of the signature, n' the number of relations in the side signature, and a' the maximal arity of the side signature. We can bound the number of relations R_θ by bounding the number of types, which amounts to bounding the number of childish instances up to isomorphism. Now, a childish instance is defined by choosing the GTGD head of Σ that we copy (a factor of $|\Sigma|$), the subset of variables on which to add side facts (a factor of $a^{O(w)}$), and a set of side facts on those elements (a factor of at most $2^{n' \cdot w^{a'}}$).

Let us now bound the number of rules. The number of rules obtained by (Instantiate) is bounded by the number of relations R_θ (bounded above), times the number of choices for the homomorphism h (i.e., w^w), times the number of possible choices for the fact in the rule head. The number of such choices can be bounded by observing that such facts are obtained by instantiating the head of a GTGD of $\widehat{\Sigma}$ (so at most $|\widehat{\Sigma}|$ choices), identifying some elements in the head (of which there are at most w, so a factor w^w to choose a homomorphism), and selecting the tuple of elements on which the fact is created (i.e., a^w).

Further, the number of rules created by (Lift) is bounded by the number of relations R_θ (bounded above), times the number of homomorphisms (i.e., w^w), times $a^{O(w)}$ for the choice of exported variables, times the number of relations $R_{\theta''}$ for the choice of head.

Note that in the bounds calculated above, if we treat a' as a constant, every exponent is either $O(w)$ or $O(n', w)$. The number of rules is a polynomial in these quantities. Hence, having fixed the arity a' of the side signature, there are indeed fixed polynomials P_1 and P_2 satisfying our claim. □

We have described the construction of the new linear constraints $\mathsf{Linearize}(\Sigma)$.

For any instance I in the original signature, we construct I^{Lin} by considering every childish instance S on domain $\vec{a}$ which is a subinstance of I and adding the fact $R_\theta(\vec{a})$ where θ is the type of S. This still respects the time bounds, because it is doable in PTIME in $|I|^w \times |\Sigma|$.

The last thing to show is that the linearization is correct:

Proposition 9.4. *Let I be an instance and I' be a superinstance formed via $\Sigma \cup \widehat{\Sigma}$ chase steps that is fact-saturated and I-deactivated for Σ. Then $I', \Sigma \models Q$ if and only if $(I')^{\mathsf{Lin}}, \mathsf{Linearize}(\Sigma) \models Q$.*

Proof. To prove the result, for an instance I'' of the signature introduced above, let $\mathsf{Delinearize}(I'')$ be the result of replacing facts over predicates R_θ by the facts corresponding to θ. Note that, using the rules created by (Instantiate), every R_θ-atom entails the corresponding θ-atoms.

The "soundness direction", from right to left, is easy to see, and it does not use fact-saturatedness of the instance I' or the fact that I' is I-deactivated. Indeed, for any linearized rule σ, suppose a chase step with σ on $(I')^{\mathsf{Lin}}$ yields I''. Then there are ordinary chase steps using rules in Σ that produce $\mathsf{Delinearize}(I'')$ from I'.

To prove the "completeness direction", we show that *for any shortcut chase sequence with Σ on I', there is a chase sequence with $\mathsf{Linearize}(\Sigma)$ whose delinearization will produce all the same facts.* Since the shortcut chase captures entailment on fact-saturated instances (from Proposition 8.5), this is sufficient to conclude.

To see this, consider the shortcut chase where each non-root node v is annotated by the type of the childish instance that this node had when it was created from its parent. Consider also the chase by the rules obtained with (Lift) in $\mathsf{Linearize}(\Sigma)$. We can observe by

a straightforward induction that there is a chase by the rules of (Lift) in $\mathsf{Linearize}(\Sigma)$ such that the tree structure on the nodes of the shortcut chase with the indicated annotations is isomorphic to the tree of facts of the form $R_\theta(\vec{x})$ created in this lifted chase.

Here, we use the fact that I' is I-deactivated in the following way. Consider the first round of non-full shortcut chase steps applied on all active triggers in I'. Consider specifically the firing of a non-full GTGD γ on a trigger τ which creates a node b in the first round of the shortcut chase. Our goal is to find a childish subinstance S of I' that we can use to replicate this firing in the linearization.

We know that the image ι of the trigger τ cannot be in I because I' is I-deactivated. So ι must involve at least one fact of $I' \setminus I$. In fact, as I' is fact-saturated, the principal fact F that guards ι must be a fact of $I' \setminus I$. But as I' was created from I by chase steps, we know that F was created in I' by firing a GTGD γ' of $\Sigma \cup \widehat{\Sigma}$. If the GTGD γ' is full, then the width bound ensures that F contains at most w distinct elements, so we can take $S := \iota$ and S is a childish subinstance of I'. If the GTGD γ' is non-full, then F was created in a new chase node together with inherited side signature facts on at most w elements, so F was part of a childish subinstance S of I' at that point; and S entails all facts guarded by F which are derived afterwards by chase steps in I', in particular those of ι.

Letting S be the childish instance formed above, let θ be the type of S. By construction of $(I')^{\mathsf{Lin}}$, in both cases there is a fact F_S for the relation R_θ on the elements of the childish instance S, and the firing of the active trigger τ in the first round of the shortcut chase can be performed in the linearization by firing a rule on F_S whose body uses relation R_θ and whose head uses the relation $R_{\theta'}$ for θ' the type of the childish instance created when firing γ on τ. This allows us to label the node b of the shortcut chase with θ'.

For subsequent rounds of the shortcut chase steps, we know that active triggers are contained in nodes that were created by a non-full chase step in the previous round and were then fully saturated; and when these nodes are created they contain a childish instance and can be labeled accordingly.

Now, the applications of the rules obtained with (Instantiate) to the linearized chase create precisely the facts in the original signature contained in these nodes, so this shows that the chase by $\mathsf{Linearize}(\Sigma)$ and the shortcut chase create the same facts.

This shows that $\mathsf{Linearize}(\Sigma)$ satisfies indeed the hypotheses of Theorem 3.6. □

We finish by putting together the steps of the proof:

Proof of Theorem 3.6. We have already explained at the end of Section 6 how we reduce to the setting where Σ strongly obeys a side signature $\mathcal{S}'$ of suitable size.

We use Theorem 7.3 to compute a childish saturation $\widehat{\Sigma}$ of Σ. The running time bound is now polynomial in $|I_0|$, $(|\Sigma| \times a)^{O(w)}$, and $2^{n' \times w^{a'}}$. Using Proposition 8.3, we perform chase steps to construct I'_0 which is Σ-fact-saturated and I_0-deactivated, with the complexity being now polynomial in the previous values and in $|I_0|^{O(w)}$. Last, we compute the linearization $(I'_0)^{\mathsf{Lin}}$ and Σ^{Lin} as explained in this section. By Claim 9.3, the overall complexity is as claimed. Further, as we explained, the obtained dependencies are linear TGDs of semi-width $\leq w$ and arity $\leq a$. Last, we know by Proposition 9.4 that the linearization is correct, i.e., $(I'_0)^{\mathsf{Lin}}$ and Σ^{Lin} emulate I'_0 and Σ, concluding the proof. □

10. Conclusion

In this work we have given finer bounds on query answering with GTGDs, based on the notion of side signature. In addition to justifying claims in prior papers, we believe it is a good example of the use of a linearization technique similar to the one from [GMP14], and also a good example of how to use the recently-developed notion of the one-pass chase. We hope that this combination could be used to get finer-grained bounds for query answering for other TGD classes, such as frontier-guarded TGDs [BLM10].

While our approach produces linearizations in the same style as [GMP14], our techniques seem to be quite different – e.g., we rely heavily on the completeness of specialized versions of the chase. It would be quite interesting to approach these bounds using the machinery and terminology of [GMP14], but we leave this for future work.

When we do not impose any width bound, but merely fix the side signature arity, we show an EXPTIME bound. Not only is the problem for this class EXPTIME-complete, but there is an EXPTIME-hardness result in [BBB13] for the case with the simplest possible fixed side signature: one unary predicate. In that sense, our EXPTIME result is optimal. We do not know whether our two results (Result 1 and 2) could be extended to the setting of multi-head TGDs or TGDs with constants in full generality (see discussion in Section 3 and see Appendix B).

Our results give new classes where OWQA is in EXPTIME and new cases where it is in NP. We do not provide larger classes where the complexity is in PSPACE: this is a bit surprising, and is one area of particular interest for future work. Indeed, the canonical example where OWQA is in PSPACE is when TGDs are linear, and our technique works by reducing to that case. Hence, it is natural to ask whether our techniques can give a PSPACE upper bound (rather than EXPTIME) for some more general classes defined via the notion of side signature.

References

[AB18] Antoine Amarilli and Michael Benedikt. When can we answer queries using result-bounded data interfaces? *PODS*, 2018. doi:10.1145/3196959.3196965.

[AB22] Antoine Amarilli and Michael Benedikt. When can we answer queries using result-bounded data interfaces? *LMCS*, 18(2), 2022. doi:10.46298/LMCS-18(2:14)2022.

[BBB13] Vince Bárány, Michael Benedikt, and Pierre Bourhis. Access patterns and integrity constraints revisited. In *ICDT*, 2013. doi:10.1145/2448496.2448522.

[BBG+22] Michael Benedikt, Maxime Buron, Stefano Germano, Kevin Kappelmann, and Boris Motik. Rewriting the infinite chase. *PVLDB*, 2022. doi:10.14778/3551793.3551851.

[BBG+24] Michael Benedikt, Maxime Buron, Stefano Germano, Kevin Kappelmann, and Boris Motik. Rewriting the infinite chase for guarded TGDs. *TODS*, 2024. doi:10.1145/3696416.

[BBJT19] Michael Benedikt, Pierre Bourhis, Louis Jachiet, and Michaël Thomazo. Reasoning about disclosure in data integration in the presence of source constraints. In *IJCAI*, 2019. doi:10.24963/IJCAI.2019/215.

[BBtC18] Vince Bárány, Michael Benedikt, and Balder ten Cate. Some model theory of guarded negation. *Journal of Symbolic Logic*, 2018. doi:10.1017/JSL.2018.64.

[BLM10] Jean-François Baget, Michel Leclère, and Marie-Laure Mugnier. Walking the decidability line for rules with existential variables. In *KR*, 2010.

[Bor06] Boris Motik. *Reasoning in description logics using resolution and deductive databases*. PhD thesis, Karlsruhe Institute of Technology, 2006.

[CF05] Balder ten Cate and Massimo Franceschet. Guarded fragments with constants. *Journal of Logic, Language and Information*, 14(3), 2005. doi:10.1007/s10849-005-5787-x.

[CGK13] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *JAIR*, 2013. doi:10.1613/JAIR.3873.

[CGL12] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *Journal of Web Semantics*, 14, 2012. doi:10.1016/J.WEBSEM.2012.03.001.

[CGLP11] Andrea Calì, Georg Gottlob, Thomas Lukasiewicz, and Andreas Pieris. A logical toolbox for ontological reasoning. *SIGMOD Record*, 40(3), 2011. doi:10.1145/2070736.2070738.

[CLR03] Andrea Calì, Domenico Lembo, and Riccardo Rosati. Query rewriting and answering under constraints in data integration systems. In *IJCAI*, 2003.

[DLN07] Alin Deutsch, Bertram Ludäscher, and Alan Nash. Rewriting queries using views with access patterns under integrity constraints. *TCS*, 371(3), 2007. doi:10.1016/J.TCS.2006.11.008.

[FKMP05] Ronald Fagin, Phokion G. Kolaitis, Renee J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. *TCS*, 336(1), 2005. doi:10.1016/J.TCS.2004.10.033.

[GMP14] Georg Gottlob, Marco Manna, and Andreas Pieris. Polynomial combined rewritings for existential rules. In *KR*, 2014.

[GMP20] Georg Gottlob, Marco Manna, and Andreas Pieris. Multi-head guarded existential rules over fixed signatures. In *KR*, 2020. doi:10.24963/KR.2020/45.

[JK84] David S. Johnson and Anthony C. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. *JCSS*, 28(1), 1984. doi:10.1016/0022-0000(84)90081-3.

[Kap19] Kevin Kappelmann. Decision procedures for guarded logics, 2019. https://arxiv.org/abs/1911.03679.

[Lib95] Leonid Libkin. *Elements of finite model theory*. Springer, 1995.

[LMPS15] Thomas Lukasiewicz, Maria Vanina Martinez, Andreas Pieris, and Gerardo I. Simari. From classical to consistent query answering under existential rules. In *AAAI*, 2015. doi:10.1609/AAAI.V29I1.9414.

[MMS79] David Maier, Alberto O. Mendelzon, and Yehoshua Sagiv. Testing implications of data dependencies. *TODS*, 4(4), 1979. doi:10.1145/320107.320115.

[One13] Adrian Onet. The chase procedure and its applications in data exchange. In *Data Exchange, Integration, and Streams*, 2013. doi:10.4230/DFU.VOL5.10452.1.

Appendix A. Proof of the semi-width result (Proposition 2.7)

In this appendix, we prove the NP bound on OWQA for bounded semi-width linear TGDs, i.e., Proposition 2.7. Recall its statement:

Proposition 2.7. *For fixed w, there is an* NP *algorithm for OWQA under linear TGDs of semi-width at most w.*

The proof presented here is the same as in [AB22, Appendix C], except that to go from IDs to linear TGDs we must change the statement of proof of Lemma A.2. The proof is otherwise identical up to minor changes.

To prove the result, let Σ be the collection of linear TGDs. We will reason about the tree-like chase sequences (see Section 5) that can be obtained starting with some instance I_0. Specifically, in this appendix, when talking about a *tree-like chase sequence*, we mean the following: we always consider *relaxed* tree-like chase sequences, where chase steps performed with full TGDs are *always* relaxed (i.e., the fact is always created in a new child node); and further we *never perform propagation steps* (they are never needed) and we *never inherit any facts* when creating fresh nodes. This is consistent with how tree-like chase sequences work in the case of IDs considered in Johnson and Klug's work [JK84].

Thanks to this assumption, in the tree-like chase sequences that we consider, the root node always contains precisely the facts of I_0, and the non-root nodes contain precisely one fact and are in one-to-one correspondence with the facts that are generated. Further,

if we fired a trigger whose image is a fact F, and this creates a fact F', then the node n' created by applying the chase step so that $T(n') = \{F'\}$ is a child of the node n such that $T(n) = \{F\}$.

Let us now consider a chase tree T within some tree-like chase proof starting with I_0. A *generated fact* in T is a fact which is not a fact of I_0. Let us now consider nodes n and n' in T, with n a strict ancestor of n'. We say n and n' are *far apart* if there are distinct generated facts F_1 and F_2 such that:

- the node n_1 corresponding to F_1 and the node n_2 corresponding to F_2 are both ancestors of n' and descendants of n;
- n_1 is an ancestor of n_2;
- F_1 and F_2 were generated by the same rule of Σ; and
- the equalities between values in positions within F_1 are exactly the same as the equalities within F_2, and any values occurring in both F_1 and F_2 occur in the same positions in F_1 and F_2.

If n and n' are not far apart, we say that they are *near*.

Given a match h of Q in the chase tree T, its *augmented image* is the closure of its image under least common ancestors, including by convention the root node. If Q has size k then this has size $\leq 2k + 1$. For any two nodes n and n' in the augmented image, we call n the *image parent of* n' if n is the lowest ancestor of n' in the augmented image.

Lemma A.1. *If Q has a match h in the final chase tree T of a tree-like chase sequence, then there is another tree-like chase sequence with final tree T', and a match h' of Q in T' with the property that if n is the image parent of n' then n and n' are near.*

Proof. We prove that given such an h and T, we can construct an h' and T' such that we decrease the sum of the depths of the violations.

If n is far apart from n', then there are witnesses F_1 and F_2 to this, corresponding to nodes n_1 and n_2 respectively. Informally, we will "pull up" the homomorphism by replacing witnesses below F_2 with witnesses below F_1. Formally, we create T' by first removing each step of the chase proof that generates a node that is a descendent of n_1. Letting T_1 be the nodes in T that do not lie below n_1, we will add nodes and the associated proof steps to T'. Let C_2 be the chase steps in T that generate a node below n_2, ordered as in T, and let T_2 be the nodes produced by these steps. We then add chase steps in T' for each chase step in C_2. More precisely, we expand T' by an induction on prefixes of C_2, building T' and a partial function m from the domains of facts in $\{n_2\} \cup T_2$ into the domain of facts associated to n_1 and its descendants in T'. The invariant is that m preserves each fact of T generated by the chase steps in C_2 we have processed thus far in the induction, and that m is the identity on any values in F_1. We initialize the induction by mapping the elements associated to n_1 to elements associated to n_2. Our assumptions on n_1 and n_2 suffice to guarantee that we can perform such a mapping satisfying the invariant. For the inductive case, suppose the next chase step s in C_2 uses linear TGD δ, firing on the fact associated to v_i in T, producing node v_{i+1}. Then we perform a step s' using δ and the fact associated to $m(v_i)$ in T'. If δ was a full TGD we do not modify m, while if it is a non-full TGD we extend m to map the generated elements of s to the corresponding elements of s'. We can thus form h' by revising $h(x)$ when $h(x)$ lies below n_1, setting $h'(x)$ to $m(h(x))$. Note that there could not have been any elements in the augmented image of h in T that hang off the path between n_1 and n_2, since n and n' were assumed to be adjacent in the augmented image and the augmented image is closed under least common ancestors.

In moving from T and h to T' and h' we reduce the sum of the depths of nodes in the image, while no new violations are created, since the image-parent relationships are preserved. □

Call a match h of Q in the chase *tight* if it has the property given in the lemma above. The *depth* of the match is the depth of the lowest node in its image. The next observation, also due to Johnson and Klug, is that when the width is bounded, tight matches cannot occur far down in the tree:

Lemma A.2. *If Σ is a set of linear TGDs of width w and the schema has arity bounded by m, then any tight match of size k has all of its nodes at depth at most $k \cdot |\Sigma| \cdot (m+w)^w$.*

Proof. We claim that the length of the path between a node n of the image of the match and its image parent n' must be at most $\Delta := |\Sigma| \cdot (m+w)^w$ Indeed, every fact on the path was created by applying a rule of Σ: choosing such a rule σ, the occurrences of variables in the head tell us which of the elements of facts created by the application of σ are necessarily equal. Specifically, the elements at positions corresponding to existential variables contain fresh values with equalities that are exactly as indicated; and the elements at exported positions contain at most w distinct values, with equalities specified by the variable occurrences plus possibly additional equalities if some of the w values are in fact equal. Thus, considering the values occurring in the fact of n' (at most m), the status of a descendant fact can be characterized by:

- the last rule used; this corresponds to a factor of $|\Sigma|$
- For each of the exported elements (at most w), knowing which are equal to elements of n' or to some different element, i.e., each such exported element is either one of the m elements of n' or some value in $\{1 \dots w\}$ used to represent the equality patterns between the elements that are not in n'; this corresponds to a factor of $(m+w)^w$

Thus, after Δ steps, there will be two elements which repeat both the rule and the configuration of the values, which would contradict tightness. Since the augmented image contains the root, this implies the bound above. □

Johnson and Klug's result, generalized from IDs to linear TGDs, follows from combining the previous two lemmas:

Proposition A.3 [JK84]**.** *For any fixed $w \in \mathbb{N}$, there is an* NP *algorithm for query containment under linear TGDs of width at most w.*

Proof. We know it suffices to determine whether there is a match in a chase proof, and the previous lemmas tell us that the portion of a chase proof required to find a match is not large. We thus guess a tree-like chase proof where the tree consists of k branches of depth at most $k \cdot |\Sigma| \cdot (m+w)^w$ for k the query size, along with a match in them, verifying the validity of the branches according to the rules of Σ. □

We now give the extension of this argument for bounded semi-width. Recall from the body that a collection of linear TGDs Σ has *semi-width* bounded by w if it can be partitioned as $\Sigma = \Sigma_1 \cup \Sigma_2$ where Σ_1 has width bounded by w and the basic position graph of Σ_2 is acyclic. An easy modification of Proposition A.3 now completes the proof of our semi-width result (Proposition 2.7):

Proof of Proposition 2.7. We revisit the argument of Lemma A.2, claiming a bound with an extra factor of $|\Sigma|$ in it. As in that argument, it suffices to show that, considering the

extended image of a tight match of Q in a chase proof, then the distance between any node n' of the extended image and its closest ancestor n is bounded, i.e., it must be at most $|\Sigma|^2 \cdot (m+w)^w$. Indeed, as soon as we apply a rule of Σ_1 along the path, at most w values are exported, and so the remaining path is bounded as before. Since Σ_2 has an acyclic basic position graph, a value in n can propagate for at most $|\Sigma_2|$ steps when using rules of Σ_2 only. Thus after at most $|\Sigma_2|$ edges in a path we will either have no values propagated (if we used only rules from Σ_2) or at most w values (if we used a rule from Σ_1). In particular, we cannot have a gap of more than $|\Sigma_2| \cdot |\Sigma| \cdot (m+w)^w$ in a tight match, establishing our desired distance bound of $|\Sigma|^2 \cdot (m+w)^w$. □

Appendix B. Supporting constants and multi-headed GTGDs

In this appendix, we make formal the claim from Section 3 that multi-headed GTGDs with constants in rule bodies can be encoded to single-headed GTGDs without constants. Thus our EXPTIME upper bound from Result 1 also applies to multi-headed GTGDs which may feature constants (provided the constants are not in rule heads).

We first formally define multi-headed GTGDs. Remember that a *single-headed TGD* was defined in Section 2 as an FO sentence of the following form: $\forall \vec{x}\ (\beta(\vec{x}) \rightarrow \exists \vec{y}\ A(\vec{x}, \vec{y}))$. A *multi-headed TGD* is defined in the same way but as: $\forall \vec{x}\ (\beta(\vec{x}) \rightarrow \exists \vec{y}\ \eta(\vec{x}, \vec{y}))$ where η is a conjunction of atoms. As in the case of single-headed GTGDs, we say that a multi-headed TGD is *guarded* if there is an atom in the body β which contains all variables occurring in β. Further, we define TGDs with *constants* (single-headed or multi-headed) by allowing atoms in TGDs to feature constants as well as variables. The constants in question can also be used in the active domain of the instance I_0 given as input to OWQA, and in the query Q given as input to OWQA. However, *we disallow constants in the head of TGDs*: we discuss at the end of the appendix why these are different.

The OWQA *problem with multi-headed TGDs with constants in rule bodies* is defined as follows: given an instance I_0, a query Q (possibly with constants), and a set of guarded TGDs Σ (which may be multi-headed, and may feature constants in rule bodies), decide whether $I_0, \Sigma \models Q$ or not.

In this appendix, we show that Result 1 also holds in this setting: for any constant number $a' \in \mathbb{N}$, if the input GTGDs Σ obey a side signature of maximal arity a', then the OWQA problem is in EXPTIME. We do this by showing that we can rewrite the input Σ to transform it to single-headed GTGDs without constants while preserving the assumption that a bounded-arity side signature is obeyed, so that we can then conclude by Result 1.

Reducing to single-headed GTGDs. We first explain how to reduce from multi-headed to single-headed GTGDs:

Lemma B.1. *Let Σ be a set of multi-headed GTGDs with constants over signature $\mathcal{S}$ obeying a side signature $\mathcal{S}'$. We can rewrite Σ in polynomial time to a set Σ' of single-headed GTGDs with constants over a signature $\mathcal{S}' \supseteq \mathcal{S}$ such that Σ' obeys side signature $\mathcal{S}'$ and such that Σ and Σ' are $\mathcal{S}$-entailment-equivalent for* OWQA.

Proof. We rewrite each GTGD of Σ separately. Let γ be a multi-headed GTGD from Σ, namely, $\gamma : \forall \vec{x}\ (\beta(\vec{x}) \rightarrow \exists \vec{y}\ \eta(\vec{x}, \vec{y}))$.

We introduce a fresh predicate P_γ in the signature, and replace γ by several TGDs.

- The single-head GTGD $\gamma' : \forall \vec{x}\ (\beta(\vec{x}) \rightarrow \exists \vec{y}\ P_\gamma(\vec{x}, \vec{y}))$

• For each atom $A(\vec{x}, \vec{y})$ in the head $\eta(\vec{x}, \vec{y})$, the full linear TGD: $\gamma_A : \forall \vec{x}\vec{y}\, P_\gamma(\vec{x}, \vec{y}) \rightarrow A(\vec{x}, \vec{y})$.

We let Σ' be the result of this transformation. The transformation is clearly in polynomial time, and the side-signature restriction is still obeyed because each new GTGD of Σ' either is linear or has the same body as a GTGD of Σ. Further, it is clear that Σ and Σ' are $\mathcal{S}$-entailment-equivalent. □

Notice that the transformation given in the proof above may increase the width of GTGDs, because it creates GTGDs whose width is as large as the maximal number of variables used in an atom of the head of a multi-headed GTGDs. While this is not a problem to generalize Result 1, it means that the same transformation cannot be used to generalize Result 2.

Eliminating constants. We next explain how to reduce to GTGDs without constants.

Lemma B.2. *Let Σ be single-headed GTGDs over signature $\mathcal{S}$ which obey a side signature $\mathcal{S}'$ and may feature constants in rule bodies. Let I_0 be an instance on $\mathcal{S}$, and let Q be a query on $\mathcal{S}$ (possibly with constants). We can rewrite $\mathcal{S}, \mathcal{S}', \Sigma, I_0, Q$ in polynomial time to:*

- *new side signature relations $\mathcal{S}''$ whose maximal arity is no greater than that of $\mathcal{S}'$;*
- *the new side signature $\mathcal{S}' \cup \mathcal{S}''$, and the new signature $\mathcal{S} \cup \mathcal{S}''$;*
- *a set Σ_2 of single-headed GTGDs without constants over the new signature $\mathcal{S} \cup \mathcal{S}''$ such that Σ' obeys the new side signature $\mathcal{S}' \cup \mathcal{S}''$;*
- *an instance I_0' over the new signature,*
- *a query Q' without constants over the new signature.*

Further, we have $I_0, \Sigma \models Q$ iff $I_0', \Sigma' \models Q'$.

Proof. We use a standard technique for mimicking constants with unary predicates in guarded logics [CF05]. For each constant c used in the GTGDs of Σ or in the query Q, we introduce a fresh unary predicate P_c which we add to the new side signature. We let the set $\mathcal{S}''$ of new side signature predicates be the set of these unary predicates, which clearly satisfies the arity bound.

We rewrite the instance I_0 to I_0' in the following way: for each constant c that occurs in the active domain of I_0, we add the new unary fact $P_c(c)$.

We rewrite the query Q to Q' in the following way: for each constant c that occurs in Q, we add a new variable x_c, replace c by x_c, and add the atom $P_c(x_c)$.

We rewrite the single-headed GTGDs Σ in the following way: for each GTGD $\forall \vec{x}\, (\beta(\vec{x}) \rightarrow \exists \vec{y}\, A(\vec{x}, \vec{y}))$, for each constant c used in the β, we introduce a new variable x_c, replace c by x_c, and add a new atom $P_c(x_c)$. The result is still a single-headed TGD; it is still guarded because the guard atom still contains all the variables (it includes all pre-existing variables as well as all of the new variables); and it now obeys the side-signature $\mathcal{S}' \cup \mathcal{S}''$ because all atoms except the principal atom of β is either in $\mathcal{S}'$ or is an atom for one of the relations P_c which is in $\mathcal{S}''$.

It is then clear that $I_0, \Sigma \models Q$ iff $I_0', \Sigma' \models Q'$. □

Notice that the transformation given in the proof above increases the number of relations in the side signature. Again, while this is not a problem to generalize Result 1, it would be a problem to generalize Result 2.

Issues with constants in TGD heads. We last discuss why the translation in Lemma B.2 cannot be used as-is when GTGDs feature constants in rule heads. The problem is that rule head with constants, e.g., $R(x, y) \rightarrow S(y, c)$, may force us to create facts involving one specific element c: this cannot be replaced by an existentially quantified variable.

One alternative translation that can be used to allow constants in rule heads is to enlarge the arity of each relation by N, where N is the number of constants used; and store the domain elements that correspond to constants in the N extra positions. However, unlike the transformations in this appendix, this would enlarge the arity of the side signature relations, so it would not preserve the constant bound on the side signature arity. We leave open the question of whether our EXPTIME bound can be extended to GTGDs with constants in the head of rules, and also leave open the question of generalizing the NP bound.