Journal of Artificial Intelligence Research 63 (2018) 191-264 Submitted 8/17; published 10/18

Query Answering with Transitive and Linear-Ordered Data

Antoine Amarilli ANTOINE.AMARILLI@TELECOM-PARISTECH.FR
LTCI, Télécom ParisTech, Université Paris-Saclay

Michael Benedikt MICHAEL.BENEDIKT@CS.OX.AC.UK
University of Oxford

Pierre Bourhis PIERRE.BOURHIS@UNIV-LILLE1.FR
CNRS CRIStAL, Université Lille 1, INRIA Lille

Michael Vanden Boom MICHAEL.VANDENBOOM@CS.OX.AC.UK
University of Oxford

Abstract

We consider entailment problems involving powerful constraint languages such as *frontier-guarded existential rules* in which we impose additional semantic restrictions on a set of distinguished relations. We consider restricting a relation to be transitive, restricting a relation to be the transitive closure of another relation, and restricting a relation to be a linear order. We give some natural variants of guardedness that allow inference to be decidable in each case, and isolate the complexity of the corresponding decision problems. Finally we show that slight changes in these conditions lead to undecidability.

1. Introduction

The *query answering problem* (or certain answer problem), abbreviated here as QA, is a fundamental reasoning problem in both knowledge representation and databases. It asks whether a query (e.g., given by an existentially-quantified conjunction of atoms) is entailed by a set of constraints and a set of facts. That is, we generalize the standard querying problem in databases to take into account not only the explicit information (the facts) but additional "implicit information" given by constraints. A common class of constraints used for QA are *existential rules*, also known as *tuple generating dependencies* (TGDs).

Although query answering is known to be undecidable for general TGDs, there are a number of subclasses that admit decidable QA, such as those based on *guardedness*. For instance, *guarded* TGDs require all variables in the body of the dependency to appear in a single body atom (the *guard*). *Frontier-guarded* TGDs (FGTGDs) relax this condition and require only that some guard atom contains the variables that occur in both head and body (Baget, Mugnier, Rudolph, & Thomazo, 2011). These classes include standard SQL referential constraints as well as important constraint classes (e.g., role inclusions) arising in knowledge representation. Guarded existential rules can be generalized to *guarded logics* that allow disjunction and negation and still enjoy decidable QA, e.g., the guarded fragment of first-order logic (GF) of Andréka, Németi, and van Benthem (1998) which captures guarded TGDs, and the guarded negation fragment (GNF) of Bárány, ten Cate, and Segoufin (2011) which captures FGTGDs.

A key challenge is to extend these decidability results to capture additional semantics of the relations that are important in practice but cannot be expressed in these classes. For example, the property that a binary relation is *transitive* or is the *transitive closure* of another relation cannot be expressed directly in guarded logics. Yet, transitive relations, such as the "part-of" relationship among components, are common in data modelling and have received significant attention (see, e.g., Horrocks & Sattler, 1999; Baget, Bienvenu, Mugnier, & Rocher, 2015). For example, using standard reasoning $Q = \exists xy(\mathsf{broken}(x) \wedge \mathsf{part\text{-}of}(x,y) \wedge \mathsf{car}(y))$, is not entailed by the set of facts $\{\mathsf{car}(c), \mathsf{motor}(m), \mathsf{sparkplug}(p), \mathsf{part\text{-}of}(p,m), \mathsf{part\text{-}of}(m,c), \mathsf{broken}(p)\}$. But enforcing that part-of is transitive—a very natural condition when modelling this situation—makes a difference, and means that Q is entailed. Hence, we would like to be able to capture these special semantics when reasoning.

A semantic restriction related to transitivity is the fact that a binary relation is a strict *linear order*: a transitive relation which is also irreflexive and total. For example, when the data stored in a relation is numerical, it may happen that the data satisfies integrity constraints involving the standard linear order $<$ on integers; e.g., for every tuple in a ternary relation the value in the first position of a binary relation is less than the value in the third. Again, the ability to reason about the additional semantics of $<$ may be crucial in inference, but it is not possible to express this in guarded logics. Query answering with additional semantic relations, e.g., linear orders or arithmetic, has been studied in the description logic and semantic web communities (Gutiérrez-Basulto, Ibáñez-García, Kontchakov, & Kostylev, 2015; Savkovic & Calvanese, 2012; Artale, Ryzhikov, & Kontchakov, 2012). However query answering with linear orders has not received much attention for arbitrary arity relations.

In this work we look at conditions that make query answering decidable. We study the three semantic restrictions above: transitivity, transitive closure, and linear ordering. We will show that there are common techniques that can be used to analyze all three cases, but also significant differences.

1.1 State of the Art

There has been extensive work on decidability results for guarded logics extended with such semantic restrictions.

We first review known results for the *satisfiability problem*, which asks whether some logical constraints are satisfiable. Ganzinger, Meyer, and Veanes (1999) showed that satisfiability is not decidable for GF when two relations are restricted to be transitive, even on arity-two signatures (i.e., with only unary and binary relations). For linear orders, Kieronski (2011) showed that GF is undecidable when three relations are restricted to be non-strict linear orders, even with only two variables (so on arity-two signatures). Otto (2001) showed that satisfiability is decidable for two-variable first-order logic with one relation restricted to be a linear order. For transitive relations, one way to regain decidability for GF satisfiability is the *transitive guards* condition introduced by Szwast and Tendera (2004): allow transitive relations *only* in guards.

We now turn to the QA *problem*, where we also consider a query and an initial set of facts. Gottlob, Pieris, and Tendera (2013) showed that query answering for GF with transitive relations only in guards is undecidable, even on arity-two signatures. Baget et al. (2015) studied QA with respect to a collection of linear TGDs (those with only a single atom in the body). They showed that the query answering problem is decidable with such TGDs and transitive relations, if the signature is arity-two or if other additional restrictions are obeyed.

The case of TGDs mentioning relations with a restricted interpretation has been studied in the database community mainly in the setting of acyclic schemas, such as those that map source data to target data. Transitivity restrictions have not been studied, but there has been work on inequalities (Abiteboul & Duschka, 1998) and TGDs with arithmetic (Afrati, Li, & Pavlaki, 2008). Due to the acyclicity assumptions, QA is decidable in these settings, and has data complexity in CoNP. The fact that the data complexity can be CoNP-hard was shown by Abiteboul and Duschka (1998), while polynomial cases were isolated by Abiteboul and Duschka (1998) with inequalities, and by Afrati et al. (2008) with arithmetic.

Query answering that features transitivity restrictions has also been studied for constraints expressed in description logics, i.e., in an arity-two setting where the signature contains unary relations (concepts) and binary relations (roles). QA is then decidable for many description logics featuring transitivity, such as $\mathcal{SHIQ}$ (Glimm, Lutz, Horrocks, & Sattler, 2008), $\mathcal{ZIQ}$, $\mathcal{ZOQ}$, $\mathcal{ZOI}$ (Calvanese, Eiter, & Ortiz, 2009), Horn-$\mathcal{SROIQ}$ (Ortiz, Rudolph, & Šimkus, 2011), OWL2 EL with the regularity restriction (Stefanoni, Motik, Krötzsch, & Rudolph, 2014), or regular-$\mathcal{EL}^{++}$ (Krötzsch & Rudolph, 2007). All of these logics are incomparable to the ones we consider. For example, the language considered by (Stefanoni et al., 2014) includes powerful features beyond transitive closure operators, such as role composition. On the other hand, it allows only arity 2 relations, and further restricts the use of inverse roles, which has a significant impact on complexity. For even more expressive description logics with transitivity, such as $\mathcal{ALCOIF}^*$ (Ortiz, Rudolph, & Šimkus, 2010) and $\mathcal{ZOIQ}$ (Ortiz de la Fuente, 2010), QA becomes undecidable. Decidability of QA is open for $\mathcal{SROIQ}$ and $\mathcal{SHOIQ}$ (Ortiz & Simkus, 2012).

1.2 Contributions

The main contribution of this work is to introduce a broad class of constraints over arbitrary-arity vocabularies where query answering is decidable even when we assert that some distinguished relations follow one of three semantics: being transitive, being the transitive closure of another relation, or being a linear order.

- We provide new results on QA with transitivity and transitive closure assertions. We show that query answering is decidable with guarded and frontier-guarded constraints, as long as these distinguished relations are *not* used as guards. We call this new kind of restriction *base-guardedness*, and similarly extend frontier-guarded to *base-frontier-guardedness*, and so forth. The base-guarded restriction is orthogonal to the prior decidable cases such as transitive guards (Szwast & Tendera, 2004) for satisfiability, or linear rules (Baget et al., 2015).

  On the one hand, we show that our restrictions make query answering decidable even with very expressive and flexible decidable logics, capable of expressing not only guarded existential rules, but also guarded rules with negation and disjunction in the head. These logics can express integrity constraints, as well as conjunctive queries and their negations. On the other hand, as a by-product of our results we obtain new query answering schemes for some previously-studied classes of guarded existential rules with extra semantic restrictions. For example, our base-frontier-guarded constraints encompass all *frontier-one TGDs* (Baget, Leclère, Mugnier, & Salvat, 2009), where at most one variable is shared between the body and head. Hence, our results imply that QA with transitivity assertions (or even transitive closure assertions) is decidable with frontier-one TGDs, which answers a question of Baget et al. (2015).

Our results are shown by arguing that it is enough to consider entailment over "tree-like" sets of facts. By representing the set of witness representations as a tree automaton, we derive upper bounds for the combined complexity of the problem. The sufficiency of tree-like examples also enables a refined analysis of *data complexity* (when the query and constraints are fixed). Further, we use a set of coding techniques to show matching lower bounds within our fragment. We also show that loosening our conditions leads to undecidability.

- We provide both upper and lower bounds on the QA problem when the distinguished relations are *linear orders*.

  We show that it is undecidable even assuming base-frontier-guardedness, so we introduce a stronger condition called *base-coveredness*: not only are distinguished relations never used as guards, they are always *covered* by a non-distinguished atom. Under these conditions, our decidable technique for QA works by "compiling away" linear order restrictions, obtaining an entailment problem without any special restrictions. The correctness proof for our reduction to classical QA again relies on the ability to restrict reasoning to sets of facts with tree-like representations. To our knowledge, these are the first decidability results for the QA problem with linear orders, and again we provide tight complexity bounds for the problem.

Both classes of results apply to the motivating scenarios for distinguished relations mentioned earlier. Our results on transitivity show that QA with distinguished relations that are transitive or are the transitive closure of a base relation is decidable for BaseGNF, the restriction of GNF that follows our base-guardedness requirement. In particular this means that in query answering with rules 1. in the *head* of a rule, or in the query, one can freely restrict some relations to be transitive, or to be the transitive closure of some other relation; 2. in the *body* of a rule, one can restrict some relations to be transitive or to be the transitive closure, provided that there is a frontier-guard available that is not restricted. In particular, we can use a transitive relation such as "part-of" (or even its transitive closure) whenever only one variable is to be exported to the head: that is in "frontier-1" rules. This latter condition holds in the translations of many classical description logics. For example, $\forall xy(\mathsf{part\text{-}of}(x,y) \wedge \mathsf{broken}(x) \rightarrow \mathsf{broken}(y))$ and $\forall p(\mathsf{sparkplug}(p) \rightarrow \exists m(\mathsf{motor}(m) \wedge \mathsf{part\text{-}of}(p,m)))$ can be rewritten in BaseGNF.

Our results on QA with linear orders show that the problem is decidable for BaseCovGNF, the base-covered version of GNF. This allows constraints that arise from data integration and data exchange over attributes with linear orders—e.g., views defined by selecting rows of a table where some order constraint involving the attributes is satisfied.

1.3 Organization

In Section 2, we formally define the query answering problems that we study, and the constraint languages that we use. We present our main decidability results on query answering with transitive data in Section 3, and with linear-ordered data in Section 4; we analyze both the combined complexity and data complexity of these decidable cases. We prove lower bounds for these problems in Section 5, and show that slight changes to the conditions lead to undecidability in Section 6. Section 6 also compares the undecidability results with prior results in the literature.

Our main results are summarized in Figure 1, and the languages that we study are illustrated in Figure 2 (please see Section 2 for the definitions). Some technical material that is not essential for understanding our main results can be found in the appendices.

Fragment	QAtr		QAtc		QAlin	
	data	**combined**	**data**	**combined**	**data**	**combined**
BaseGNF	coNP-c	2EXP-c	coNP-c	2EXP-c	undecidable	
BaseCovGNF	coNP-c	2EXP-c	coNP-c	2EXP-c	coNP-c	2EXP-c
BaseFGTGD	in coNP	2EXP-c	coNP-c	2EXP-c	undecidable	
BaseCovFGTGD	P-c	2EXP-c	coNP-c	2EXP-c	coNP-c	2EXP-c

Figure 1: Summary of QA results. On the rows that concern base-covered fragments, the queries are also assumed to be base-covered. For complexity class X, we write "X-c" for "X-complete". Please refer to Sections 3 and 4 for upper bounds, Section 5 for lower bounds, and Section 6 for undecidability results.

2. Preliminaries

We work on a *relational signature* σ, where each relation $R \in \sigma$ has an associated *arity* written $\mathrm{arity}(R)$; we write $\mathrm{arity}(\sigma) := \max_{R\in\sigma} \mathrm{arity}(R)$. A *fact* $R(\vec{a})$, or *R-fact*, consists of a relation $R \in \sigma$ and elements $\vec{a}$, with $|\vec{a}| = \mathrm{arity}(R)$. $\vec{a}$ is the *domain* of the fact. Queries and constraints will be evaluated over a (finite or infinite) set of facts over σ. We will often use $\mathcal{F}$ to denote a set of facts. We write $\mathrm{elems}(\mathcal{F})$ for the set of elements that appear as arguments in the facts in $\mathcal{F}$. We also refer to this as the *domain of* $\mathcal{F}$.

We consider *constraints* and *queries* given in fragments of first-order logic with equality (FO) without constants. Given a set of facts $\mathcal{F}$ and a sentence φ in FO, we talk of $\mathcal{F}$ *satisfying* φ in the usual way. The *size* of φ, written $|\varphi|$, is defined to be the number of symbols in φ.

A *conjunctive query* (CQ) is a first-order formula of the form $\exists\vec{x}\,\varphi$ for φ a conjunction of atomic formulas using equality or a relation from σ. Likewise, a *union of conjunctive queries* (UCQs) is a disjunction of CQs. We will only use queries that are *Boolean* CQs or UCQs (i.e., CQs or UCQs with no free variables).

2.1 Problems Considered

Given a *finite* set of facts $\mathcal{F}_0$, constraints Σ and query Q (given as FO sentences), we say that $\mathcal{F}_0$ and Σ *entail* Q if for every $\mathcal{F} \supseteq \mathcal{F}_0$ satisfying Σ (including infinite $\mathcal{F}$), we have that $\mathcal{F}$ satisfies Q. This amounts to asking whether $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ is unsatisfiable, over all finite and infinite sets of facts. We write $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ for this decision problem, called the *query answering* problem.

In this paper, we study the QA problem when imposing semantic constraints on some *distinguished* relations. We thus work with signatures of the form $\sigma := \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$, where $\sigma_{\mathcal{B}}$ is the *base signature* (its relations are the *base* relations), and $\sigma_{\mathcal{D}}$ is the *distinguished* signature (with *distinguished* relations), and $\sigma_{\mathcal{B}}$ and $\sigma_{\mathcal{D}}$ are disjoint. All distinguished relations are required to be binary, and they will be assigned special semantics.

We study three kinds of special semantics:

- We say $\mathcal{F}_0, \Sigma$ *entails* Q *over transitive relations*, and write $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ for the corresponding problem, if there is no set of facts $\mathcal{F}$ where $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ holds and each relation $R_i^+ \in \sigma_{\mathcal{D}}$ is *transitive*.[1]

- We say $\mathcal{F}_0, \Sigma$ *entails* Q *over transitive closure*, and write $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ for this problem, if there is no set of facts $\mathcal{F}$ where $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ holds and for each relation $R_i \in \sigma_{\mathcal{B}}$, the relation $R_i^+ \in \sigma_{\mathcal{D}}$ is interpreted as the transitive closure of R_i.

- We say $\mathcal{F}_0, \Sigma$ *entails* Q *over linear orders*, and write $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ for this problem, if there is no set of facts $\mathcal{F}$ where $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ holds and each relation $<_i \in \sigma_{\mathcal{D}}$ is a strict linear order on the elements of $\mathcal{F}$.

Example 2.1. Referring back to the example used in the introduction, we consider the query $Q = \exists xy(\mathsf{broken}(x) \wedge \mathsf{part\text{-}of}(x, y) \wedge \mathsf{car}(y))$, and the set of facts $\mathcal{F}_0 = \{\mathsf{car}(c), \mathsf{motor}(m), \mathsf{sparkplug}(p), \mathsf{part\text{-}of}(p, m), \mathsf{part\text{-}of}(m, c), \mathsf{broken}(p)\}$. The query Q is not entailed by $\mathcal{F}_0$ in general, but it is entailed in both QAtr, QAtc, and QAlin when the distinguished relation part-of is asserted to be transitive.

The difference in semantics between QAtr and QAtc can be exemplified with the rule

$$\mathsf{same\text{-}supplier}(x, y), \mathsf{part\text{-}of}^+(x, y), \neg\mathsf{part\text{-}of}(x, y) \rightarrow \mathsf{indirect\text{-}pair}(x, y),$$

which applies to all pairs of objects x, y produced by the same supplier such that x is *indirectly* a part of y. This rule can be expressed in the QAtc setting, whereas in QAtr we cannot distinguish between part-of and its transitive closure part-of$^+$.

As for the semantics of QAlin, it differs from both QAtr and QAtc. Consider the set of facts $\mathcal{F}_0' = \{\mathsf{motor}(m), \mathsf{sparkplug}(p), \mathsf{prod\text{-}date}(m, d), \mathsf{prod\text{-}date}(p, d')\}$ and the UCQ Q' with the following disjuncts:

$$\exists xyzz'(\mathsf{motor}(x) \wedge \mathsf{sparkplug}(y) \wedge \mathsf{prod\text{-}date}(x, z) \wedge \mathsf{prod\text{-}date}(y, z') \wedge z < z')$$
$$\exists xyzz'(\mathsf{motor}(x) \wedge \mathsf{sparkplug}(y) \wedge \mathsf{prod\text{-}date}(x, z) \wedge \mathsf{prod\text{-}date}(y, z') \wedge z' < z)$$
$$\exists xyz(\mathsf{motor}(x) \wedge \mathsf{sparkplug}(y) \wedge \mathsf{prod\text{-}date}(x, z) \wedge \mathsf{prod\text{-}date}(y, z)).$$

The query Q is entailed in QAlin over $\mathcal{F}_0'$ where $<$ is asserted to be a total order, but it is not entailed in either QAtr or QAtc.

We now define the constraint languages for which we study these QA problems. We will also give some examples of sentences in these languages in Example 2.2.

2.2 Dependencies

The first constraint languages that we study are restricted classes of *tuple-generating dependencies* (TGDs). A TGD is an FO sentence τ of the form $\forall \vec{x} \left(\bigwedge_i \gamma_i(\vec{x}) \rightarrow \exists \vec{y} \bigwedge_i \rho_i(\vec{x}, \vec{y}) \right)$ where $\bigwedge_i \gamma_i$ and $\bigwedge_i \rho_i$ are non-empty conjunctions of atoms, respectively called the *body* and *head* of τ.

We will be interested in TGDs that are *guarded* in various ways. A *guard* for a tuple $\vec{x}$ of variables, or for an atom $A(\vec{x})$, is an atom from σ or an equality using (at least) every variable

1. Note that we work with *transitive* relations, which may not be *reflexive*, unlike, e.g., R^* roles in $\mathcal{ZOIQ}$ description logics (Calvanese et al., 2009). This being said, all our results extend with the same complexity to relations that are both reflexive and transitive. Please refer to Section 3.3 for more information.

in $\vec{x}$. That is, an atom where every variable of $\vec{x}$ appears as an argument. For example, $R(z, y)$, $C(y, w, z)$, and $y = z$ are all guards for $\vec{x} = (y, z)$. In this work, we will be particularly interested in *base-guards* (sometimes denoted $\sigma_{\mathcal{B}}$-guards), which are guards coming from the base relations in $\sigma_{\mathcal{B}}$ or equality.

A *frontier-guarded TGD* or FGTGD (Baget et al., 2011) is a TGD τ whose body contains a guard for the *frontier variables*, i.e., the variables that occur in both head and body. We introduce the *base frontier-guarded TGDs* (BaseFGTGDs) as the TGDs with a *base frontier guard*, i.e., an equality or $\sigma_{\mathcal{B}}$-atom including all the frontier variables. We allow equality atoms $x = x$ to be guards, so BaseFGTGD subsumes *frontier-one TGDs* (Baget et al., 2011), which have one frontier variable. We also introduce the more restrictive class of *base-covered frontier-guarded TGDs* (BaseCovFGTGD): they are the BaseFGTGDs where, for every $\sigma_{\mathcal{D}}$-atom A in the body, there is a base-guard A' of A in the body; note that this time A' may be different for each A.

Inclusion dependencies (ID) are an important special case of frontier-guarded TGDs used in many applications. An ID is a FGTGD of the form $\forall \vec{x}\, R(\vec{x}) \rightarrow \exists \vec{y}\, S(\vec{x}, \vec{y})$, i.e., where the body and head contain a single atom, and where we further impose that no variable occurs twice in the same atom. A *base inclusion dependency* (BaseID) is an ID where the body atom is in $\sigma_{\mathcal{B}}$, so the body atom serves as the base-guard for the frontier variables, and the constraint is trivially base-covered.

2.3 Guarded Logics

Moving beyond TGDs, we also study constraints coming from *guarded logics*. In particular, the *guarded negation fragment* (GNF) over a signature σ is the fragment of FO given by the grammar

$$\varphi ::= A(\vec{x}) \mid x = y \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists \vec{x}\, \varphi \mid \alpha \wedge \neg\varphi$$

where A ranges over relations in σ and α is a guard for the free variables in $\neg\varphi$.

Note that since a guard α in $\alpha \wedge \neg\varphi$ is allowed to be an equality, GNF can express all formulas of the form $\neg\varphi$ when φ has at most one free variable. In particular, if a sentence φ is in GNF, then $\neg\varphi$ is expressible in GNF, as $\exists x\, x = x \wedge \neg\varphi$.

The use of these "equality-guards" is convenient in the proofs. But in the presentation of examples within the paper, we do not wish to write out these "dummy guards", and thus as a convention we allow in examples unguarded subformulas $\neg\varphi$ where φ has at most one free variable.

GNF can express all FGTGDs since an FGTGD of the form $\forall \vec{x} (\bigwedge \gamma_i \rightarrow \exists \vec{y} \bigwedge \rho_i)$ can be written in BaseGNF as $\neg \exists \vec{x} (\bigwedge \gamma_i \wedge \alpha \wedge \neg \exists \vec{y} \bigwedge \rho_i)$ where α is the guard for the frontier variables in $\bigwedge \gamma_i$. It can also express non-TGD constraints and UCQs. For instance, as it allows disjunction, GNF can express *disjunctive inclusion dependencies*, DIDs (Bourhis, Morak, & Pieris, 2013), which generalize IDs: a DID is a first-order sentence of the form $\forall \vec{x}\, R(\vec{x}) \rightarrow \bigvee_{1 \leq i \leq n} \exists \vec{y_i}\; S_i(\vec{x}, \vec{y_i})$ such that, for every $1 \leq i \leq n$, the sentence $\forall \vec{x}\, R(\vec{x}) \rightarrow \exists \vec{y_i}\; S_i(\vec{x}, \vec{y_i})$ is an ID. In particular, any ID is a DID, as is seen by taking $n = 1$ in the disjunction.

In this work, we introduce the *base-guarded negation fragment* BaseGNF over σ: it is defined like GNF, but requires *base-guards* instead of guards. The *base-covered guarded negation fragment* BaseCovGNF over σ consists of BaseGNF formulas such that every $\sigma_{\mathcal{D}}$-atom A that appears negatively (i.e., under the scope of an odd number of negations) appears in conjunction with a base-guard for its variables. This condition is designed to generalize BaseCovFGTGDs; indeed, any BaseCovFGTGD can be expressed in BaseCovGNF.

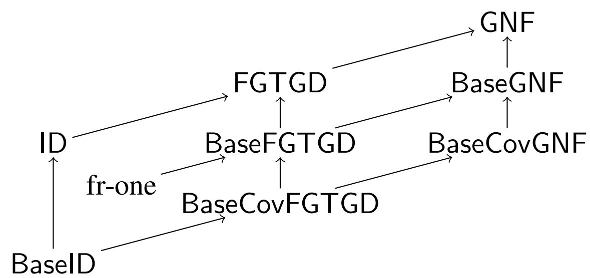

Figure 2: Taxonomy of fragments

We call a CQ Q *base-covered* if, for each $\sigma_{\mathcal{D}}$-atom A in Q, there is a base-guard A' of A in Q. A UCQ is *base-covered* if each disjunct is. Note that every base-covered UCQ can easily be rewritten in BaseCovGNF.

We illustrate the different constraint languages and queries by giving a few examples.

Example 2.2. Consider a signature with a binary base relation B, a ternary base relation C, and a distinguished relation D.

- $\forall xyz\big((B(x,y) \wedge B(y,z)) \rightarrow D(x,z)\big)$ is a TGD, but is not a FGTGD since the frontier variables x, z are not guarded. It cannot even be expressed in GNF.
- $\forall xy\big(D(x,y) \rightarrow B(x,y)\big)$ is an ID, hence a FGTGD. It is not a BaseID or even in BaseGNF, since the frontier variables are not base-guarded.
- $\forall xyz\big((B(z,x) \wedge D(x,y) \wedge D(y,z)) \rightarrow D(x,z)\big)$ is a BaseFGTGD. However, it is not a BaseCovFGTGD since there are no base atoms in the body to cover x, y and y, z.
- $\exists wxyz\big(D(w,x) \wedge D(x,y) \wedge D(y,z) \wedge D(z,w) \wedge C(w,x,y) \wedge C(y,z,w)\big)$ is a base-covered CQ.
- $\exists xy\big(B(x,y) \wedge \neg(D(x,y) \wedge D(y,x)) \wedge (D(x,y) \vee D(y,x))\big)$ cannot be rewritten as a TGD. But it can be rewritten in BaseCovGNF as

$$\begin{aligned}\exists xy\big(\quad & \big[(B(x,y) \wedge \neg D(x,y)) \vee (B(x,y) \wedge \neg D(y,x))\big] \\ & \wedge \big[(B(x,y) \wedge D(x,y)) \vee (B(x,y) \wedge D(y,x))\big] \qquad \big).\end{aligned}$$

2.4 Normal Form

The fragments of GNF that we consider can be converted into a normal form that is related to the GN normal form introduced in the original paper on GNF (Bárány et al., 2011). The idea is that GNF formulas can be seen as being built up by nesting UCQs using guarded negation. We introduce this normal form here, and discuss related notions that we will use in the proofs.

The *normal form for* BaseGNF over σ can be defined recursively as the formulas of the form $\delta[Y_1 := \alpha_1 \wedge \neg\varphi_1, \ldots, Y_n := \alpha_n \wedge \neg\varphi_n]$ where

- δ is a UCQ over signature $\sigma \cup \{Y_1, \ldots, Y_n\}$ for some fresh relations $Y_1, \ldots, Y_n$,
- $\varphi_1, \ldots, \varphi_n$ are in normal form BaseGNF over σ, and

- $\alpha_1, \ldots, \alpha_n$ are base-guards for the free variables in $\varphi_1, \ldots, \varphi_n$ such that the number of free variables in each $\alpha_i \wedge \neg\varphi_i$ matches the arity of Y_i, and
- $\delta[Z := \psi]$ is the result of replacing every occurrence of $Z(\vec{x})$ in δ with $(\psi(\vec{x}))$.

The base case of this recursive definition is a UCQ over σ (take $n = 0$ above).

In other words, formulas in normal form BaseGNF are built up from UCQs (over both base and distinguished relations) using base-guarded negation. We also refer to these as *UCQ-shaped formulas*. A single disjunct of a UCQ-shaped formula is called a *CQ-shaped formula*. Note that an atomic formula can be seen as a simple UCQ with no disjunction, projection or negation.

The *normal form for* BaseCovGNF over σ consists of normal form BaseGNF formulas such that for every CQ-shaped subformula $\delta[Y_1 := \alpha_1 \wedge \neg\varphi_1, \ldots, Y_n := \alpha_n \wedge \neg\varphi_n]$ that appears negatively (in the scope of an odd number of negations), and for every distinguished atom α' that appears as a conjunct in δ, there must be some base-guard for the free variables of α' that appears in $\alpha_1, \ldots, \alpha_n$ or as a conjunct in δ. Formulas in normal form BaseCovGNF might not syntactically satisfy the condition that every distinguished atom appears in (direct) conjunction with a base atom using its variables, but it can always be converted into one satisfying this condition with only a linear blow-up in size by duplicating guards: e.g., if D is the only distinguished relation, $\neg\exists xyz(D(x,y) \wedge D(y,z) \wedge C(x,y,z))$ could be converted to $\neg\exists xyz(C(x,y,z) \wedge D(x,y) \wedge C(x,y,z) \wedge D(y,z))$. We allow this slightly more relaxed definition for normal form BaseCovGNF since it is more natural when talking about formulas built up using UCQ-shaped subformulas.

We revisit some of the sentences from Example 2.2 to see how to rewrite them in normal form.

Example 2.3. The BaseFGTGD $\forall xyz\big((B(z,x) \wedge D(x,y) \wedge D(y,z)) \rightarrow D(x,z)\big)$ can be expressed in normal form BaseGNF as $\neg\exists xyz\big(D(x,y) \wedge D(y,z) \wedge (B(z,x) \wedge \neg D(x,z))\big)$. We now explain how this is built following the definition of normal form BaseGNF. We first build the inner CQ-shaped formula by taking the CQ $\exists xyz\big(D(x,y) \wedge D(y,z) \wedge Y(x,z)\big)$ and substituting $B(z,x) \wedge \neg D(x,z)$ for $Y(x,z)$. We can then build the final formula by substituting $\exists xyz\big(D(x,y) \wedge D(y,z) \wedge (B(z,x) \wedge \neg D(x,z))\big)$ for the 0-ary relation Z in $\neg Z$.

The sentence $\exists xy\big(B(x,y) \wedge \neg(D(x,y) \wedge D(y,x)) \wedge (D(x,y) \vee D(y,x))\big)$ can be expressed in normal form BaseCovGNF as $\delta[Y_1 := B(x,y) \wedge \neg D(x,y), Y_2 := B(x,y) \wedge \neg D(y,x)]$ where

$$\begin{aligned}\delta = \quad & \exists xy(B(x,y) \wedge D(x,y) \wedge Y_1(x,y)) \vee \exists xy(B(x,y) \wedge D(y,x) \wedge Y_1(x,y)) \\ & \vee \exists xy(B(x,y) \wedge D(x,y) \wedge Y_2(x,y)) \vee \exists xy(B(x,y) \wedge D(y,x) \wedge Y_2(x,y)).\end{aligned}$$

2.4.1 WIDTH, CQ-RANK, AND NEGATION DEPTH

For φ in normal form BaseGNF, we define the *width* of φ to be the maximum number of free variables in any subformula of φ. Note that by reusing variable names, a formula of width k can always be written using only k variable names. The *CQ-rank* of φ is the maximum number of conjuncts in any CQ-shaped subformula $\exists\vec{x}(\bigwedge \gamma_i)$. The *negation depth* is the maximal nesting depth of negations in the syntax tree. These parameters will be important in later proofs.

2.5 Conversion into Normal Form

Observe that formulas in BaseFGTGD or BaseCovFGTGD are of the form $\forall\vec{x}(\bigwedge \gamma_i \rightarrow \exists\vec{y} \bigwedge \rho_i)$ and so can be naturally written in normal form BaseGNF or BaseCovGNF as $\neg\exists\vec{x}(\bigwedge \gamma_i \wedge (\alpha \wedge \neg\exists\vec{y} \bigwedge \rho_i))$

where α is the base-guard from $\bigwedge \gamma_i$ for the frontier-variables. In this case, there is a linear blow-up in the size. In general, BaseGNF formulas can be converted into normal form, but with an exponential blow-up in size:

Proposition 2.4. *Let φ be a formula in* BaseGNF*. We can construct an equivalent φ' in normal form in* EXPTIME *such that (i) $|\varphi'|$ is at most exponential in $|\varphi|$, (ii) the width of φ' is at most $|\varphi|$, (iii) the CQ-rank of φ' is at most $|\varphi|$, (iv) if φ is in* BaseCovGNF*, then φ' is in normal form* BaseCovGNF.

Proof sketch. The conversion works by using the rewrite rules given by Bárány et al. (2011):

$$\exists x(\theta \vee \psi) \rightsquigarrow (\exists x\theta) \vee (\exists x\psi) \qquad\qquad \theta \wedge (\psi \vee \chi) \rightsquigarrow (\theta \wedge \psi) \vee (\theta \wedge \chi)$$
$$\exists x(\theta) \wedge \psi \rightsquigarrow \exists x'(\theta[x'/x] \wedge \psi) \text{ where } x' \text{ is fresh}$$

It is straightforward to check the bounds on the size, width, and CQ-rank after performing this rewriting. Coveredness is also preserved since (i) the rewrite rules do not change the polarity of any subformulas, so any distinguished atom in φ' that appears negatively—and hence requires a base-guard—can be associated with a distinguished atom in φ that appears negatively and appears in conjunction with a base-guard for its free variables, and (ii) the rewrite rules do not separate a conjunction of two atoms, so the base-guard that appears in conjunction with a distinguished atom in φ is propagated to any occurrence of this distinguished atom in φ'. □

2.6 Automata-Related Tools

In Section 3, we will use automata running on infinite binary trees, so we briefly recall some definitions and key properties. Our presentation is partially adapted from Benedikt, Bourhis, and Vanden Boom (2016); for more background please refer to Appendix B.1 or surveys by Thomas (1997) and Löding (2011). In particular, we will need to use 2-way automata that can move both up and down as they process the tree, so we highlight some less familiar properties about the relationship between 2-way and 1-way versions of these automata. For readers not interested in the details of the automaton construction in Section 3.2, this section can be skipped.

2.6.1 TREES

The input to the automata will be infinite full binary trees T over some *tree signature* Γ consisting of a set of unary relations. That is, each node v has exactly two children (one left child, and one right), and has a label $T(v) \in \mathcal{P}(\Gamma)$ that indicates the set of unary relations that hold at v. We will assume that Γ always includes unary relations left and right, and the label of each node correctly indicates whether the node is the left or right child of its parent (with the root being the unique node with neither left nor right in its label). We will also identify each node in a binary tree with a finite string over $\{0, 1\}$, with ϵ identifying the root, and $u0$ and $u1$ identifying the left child and right child of node u.

2.6.2 TREE AUTOMATA

We define a set of *directions* Dir $:= \{\mathsf{left}, \mathsf{right}, \mathsf{up}, \mathsf{stay}\}$, and write $\mathcal{B}^+(X)$ for any set X to denote the set of positive Boolean formulas over X. A *2-way alternating parity tree automaton* (2APT) $\mathcal{A}$ is then a tuple $\langle \Gamma, Q, q_0, \delta, \Omega \rangle$ where Γ is a tree signature, Q is a finite set of states, $q_0 \in Q$ is the initial state, $\delta : Q \times \mathcal{P}(\Gamma) \to \mathcal{B}^+(\mathrm{Dir} \times Q)$ is the transition function, and $\Omega : Q \to P$ is the priority function with a finite set of *priorities* $P \subseteq \mathbb{N}$. Intuitively, the transition function δ maps a state and a set of unary relations from Γ to a positive Boolean formula over $\mathrm{Dir} \times Q$ that indicates possible next moves for the automaton.

Running the automaton $\mathcal{A}$ on some input tree T is best thought of in terms of an *acceptance game* or *membership game* (see Löding, 2011 for more information). The positions in the game are of the form $(q, v) \in Q \times T$. In position (q, v), Eve and Adam play a subgame based on $\delta(q, T(v))$, with Eve resolving disjunctions and Adam resolving conjunctions until an atom (d, q') in $\delta(q, T(v))$ is selected. Then the game continues from position (q', v') where v' is the node in direction d from v; in particular $v' := v$ if $d = \mathsf{stay}$. For example, if $\delta(q, T(v)) = (\mathsf{up}, s_1) \vee ((\mathsf{stay}, s_2) \wedge (\mathsf{right}, s_3))$, then the acceptance game starting from (q, v) would work as follows: Eve would select one of the disjuncts; if she selects the first disjunct then the game would continue from (s_1, u) where u is the parent of v, otherwise, Adam would choose one of the conjuncts and the game would continue from (s_2, v) or $(s_3, v1)$ depending on his choice.

A play $(q_0, v_0)(q_1, v_1) \dots$ is a sequence of positions in such a game. The play is winning for Eve if it satisfies the *parity condition*: the maximum priority occurring infinitely often in $\Omega(q_0)\Omega(q_1) \dots$ is even. A *strategy* for Eve is a function that, given the history of the play and the current position (q, v) in the game, determines Eve's choices in the subgame based on $\delta(q, T(v))$. Note that we allow the automaton to be started from arbitrary positions in the tree, rather than just the root. We say that $\mathcal{A}$ *accepts* T *starting from* v_0 if Eve has a strategy such that all plays consistent with the strategy starting from (q_0, v_0) are winning. $L(\mathcal{A})$ denotes the *language* of trees accepted by $\mathcal{A}$ starting from the root.

A 1-way alternating automaton is an automaton that processes the tree in a top-down fashion, using only directions left and right. A (1-way) nondeterministic automaton is a 1-way alternating automaton such that every transition function formula is of the form $\bigvee_j (\mathsf{left}, q_j) \wedge (\mathsf{right}, r_j)$. We call such automata *1-way nondeterministic parity tree automata*, and write it $\mathsf{1NPT}$.

We review some closure properties of these automata in Appendix B.

2.6.3 CONNECTIONS BETWEEN 2-WAY AND 1-WAY AUTOMATA

It was shown by Vardi (1998) that every $\mathsf{2APT}$ can be converted to an equivalent $\mathsf{1NPT}$, with an exponential blow-up.

Theorem 2.5 (Vardi, 1998)**.** *Let $\mathcal{A}$ be a $\mathsf{2APT}$. We can construct a $\mathsf{1NPT}$ $\mathcal{A}'$ such that $L(\mathcal{A}) = L(\mathcal{A}')$. The number of states of $\mathcal{A}'$ is exponential in the number of states of $\mathcal{A}$, and the number of priorities of $\mathcal{A}'$ is polynomial in the number of states and priorities of $\mathcal{A}$. The running time of the algorithm is polynomial in the size of $\mathcal{A}'$, and is hence in $\mathsf{EXPTIME}$ in the input.*

1-way nondeterministic tree automata can be seen as a special case of 2-way alternating automata, so the previous theorem shows that $\mathsf{1NPT}$ and $\mathsf{2APT}$ are equivalent, in terms of their ability to recognize trees starting from the root.

We need another conversion from 1-way nondeterministic automata to 2-way alternating automata, that we call *localization*. This process takes a 1-way nondeterministic automaton that runs on trees with extra information about membership in certain relations (annotated on the tree), and converts that automaton to an equivalent 2-way alternating automaton that operates on trees without these annotations, under the assumption that these relations hold only locally at the position the 2-way automaton is launched from. A similar localization theorem is present in prior work (Bourhis, Krötzsch, & Rudolph, 2015; Benedikt et al., 2016). We sketch the idea here, and provide more details about the construction in Appendix B.

Theorem 2.6. *Let $\Gamma' := \Gamma \cup \{P_1, \ldots, P_j\}$. Let $\mathcal{A}'$ be a* $\mathsf{1NPT}$ *on Γ'-trees. We can construct a* $\mathsf{2APT}$ *$\mathcal{A}$ on Γ-trees such that for all Γ-trees T and nodes v in the domain of T,*

$$\mathcal{A}' \text{ accepts } T' \text{ from the root iff } \mathcal{A} \text{ accepts } T \text{ from } v$$

where T' is the Γ'-tree obtained from T by setting $P_1^{T'} = \cdots = P_j^{T'} = \{v\}$. The number of states of $\mathcal{A}$ is linear in the number of states of $\mathcal{A}'$, the number of priorities of $\mathcal{A}$ is linear in the number of priorities of $\mathcal{A}'$, and the overall size of $\mathcal{A}$ is linear in the size of $\mathcal{A}'$. The running time is polynomial in the size of $\mathcal{A}'$, and hence is in PTIME.

Proof sketch. $\mathcal{A}$ simulates $\mathcal{A}'$ by guessing in a backwards fashion an initial part of a run of $\mathcal{A}'$ on the path from v to the root and then processing the rest of the tree in a normal downwards fashion. The subtlety is that the automaton $\mathcal{A}$ is reading a tree without valuations for $P_1, \ldots, P_j$ so once the automaton leaves node v, if it were to cross this node again, it would be unable to correctly simulate $\mathcal{A}'$. To avoid this issue, we only send downwards copies of the automaton in directions that are not on the path from the root to v. □

3. Decidability Results for Transitivity

We are now ready to explore query answering for $\mathsf{BaseGNF}$ when the distinguished relations are transitively closed or are the transitive closure of certain base relations. We show that these query answering problems can be reduced to tree automata emptiness testing.

3.1 Deciding QAtc Using Automata

We first consider QAtc, where $\sigma_{\mathcal{B}}$ includes binary relations $R_1, \ldots, R_n$, and $\sigma_{\mathcal{D}}$ consists of binary relations $R_1^+, \ldots, R_n^+$ such that R_i^+ is the transitive closure of R_i for each i.

Theorem 3.1. *We can decide* $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ *in* $\mathsf{2EXPTIME}$*, where $\mathcal{F}_0$ ranges over finite sets of facts, Σ over* $\mathsf{BaseGNF}$ *constraints, and Q over UCQs. In particular, this holds when Σ consists of* $\mathsf{BaseFGTGD}$*s.*

In order to prove Theorem 3.1, we give a decision procedure to determine whether $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ is unsatisfiable, when R_i^+ is interpreted as the transitive closure of R_i. When $\Sigma \in \mathsf{BaseGNF}$ and Q is a Boolean UCQ, then $\Sigma \wedge \neg Q$ is in $\mathsf{BaseGNF}$. So it suffices to show that $\mathsf{BaseGNF}$ satisfiability is decidable in $\mathsf{2EXPTIME}$, when properly interpreting R_i^+.

As mentioned in the introduction, our proofs rely heavily on the fact that in query answering problems for the constraint languages that we study, one can restrict to sets of facts that have a “tree-like” structure. We now make this notion precise. A *tree decomposition* of $\mathcal{F}$ consists of a directed graph (T, Child), where $\mathsf{Child} \subseteq T \times T$ is the edge relation, and of a labelling function λ associating each node of T to a set of elements of $\mathcal{F}$ and a set of facts over these elements, called the *bag* of that node. We impose the following conditions on tree decompositions: (i) (T, Child) is a tree; (ii) each fact of $\mathcal{F}$ must be in the image of λ; (iii) for each element $e \in \mathrm{elems}(\mathcal{F})$, the set of nodes that is associated with e by λ forms a connected subtree of T. A tree decomposition is

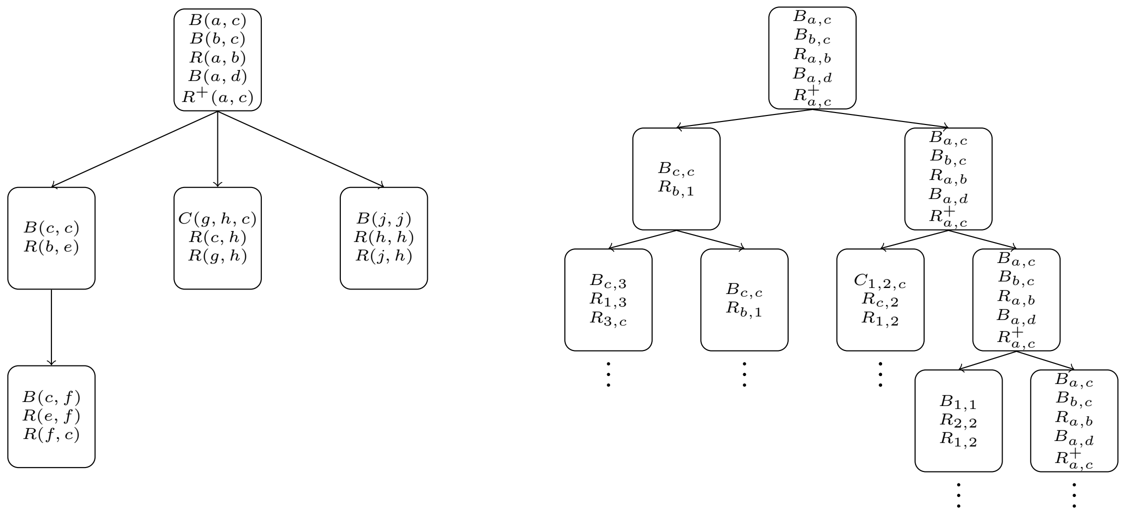

Figure 3: An $\mathcal{F}_0$-rooted tree decomposition, and part of its encoding (see Examples 3.2 and 3.5)

$\mathcal{F}_0$-*rooted* if the root node is associated with $\mathcal{F}_0$. It has *width* $k-1$ if each bag other than the root is associated with at most k elements.

For a number k, a sentence φ over σ is said to have *transitive-closure friendly k-tree-like witnesses* if: for every finite set of facts $\mathcal{F}_0$, if there is a set of facts $\mathcal{F}$ (finite or infinite) extending $\mathcal{F}_0$ with additional $\sigma_{\mathcal{B}}$-facts such that $\mathcal{F}$ satisfies φ when each R^+ is interpreted as the transitive closure of R, then there is such an $\mathcal{F}$ that has an $\mathcal{F}_0$-rooted $(k-1)$-width tree decomposition with countable branching (i.e., each node has a countable number of children). Note that, in such an $\mathcal{F}$, the only $\sigma_{\mathcal{D}}$-facts explicitly appearing in $\mathcal{F}$ are from $\mathcal{F}_0$, and we require of $\mathcal{F}$ that these distinguished facts are actually part of the transitive closure of the corresponding base relation. Other $\sigma_{\mathcal{D}}$-facts may be implied by $\sigma_{\mathcal{B}}$-facts, and both the explicit and implicit $\sigma_{\mathcal{D}}$-facts must be considered when reasoning about φ. However, we emphasize that besides the $\sigma_{\mathcal{D}}$-facts in $\mathcal{F}_0$, there are no $\sigma_{\mathcal{D}}$-facts appearing in the tree decomposition—the explicit inclusion of such $\sigma_{\mathcal{D}}$-facts could make it impossible to find a k-tree-like witness.

Example 3.2. Let $\mathcal{F}_0 = \{B(a,c), B(b,c), R(a,b), B(a,d), R^+(a,c)\}$. Figure 3 shows an $\mathcal{F}_0$-rooted tree decomposition for a set $\mathcal{F}$ of facts extending $\mathcal{F}_0$. The width of the tree decomposition is 2. $\mathcal{F}$ satisfies $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ where

$$\Sigma = \{\exists xyz(R(x,y) \wedge R(y,z) \wedge \neg R^+(z,y))\}$$
$$Q = \exists xy(B(x,y) \wedge R(x,y)).$$

For instance, $R^+(a,c) \in \mathcal{F}_0$ is satisfied in $\mathcal{F}$ because of the chain of facts $R(a,b)$, $R(b,e)$, $R(e,f)$, $R(f,c)$. Observe that the only transitive facts that explicitly appear in the tree decomposition are at the root, but other transitive facts are implied by the facts (e.g., $R^+(b,f)$) and must be taken into account when reasoning about $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$.

We can show that $\mathsf{BaseGNF}$ sentences have these transitive-closure friendly k-tree-like witnesses for an easily computable k. The proof uses a standard technique, involving an unravelling based on a form of "guarded negation bisimulation", so we defer the proof of this result to Appendix A.1. The result does not follow directly from the fact that GNF has tree-like witnesses (Bárány et al., 2011) since we must show that this unravelling preserves BaseGNF sentences even when interpreting each distinguished R^+ as the transitive closure of R, rather than just preserving GNF sentences without these special interpretations. However, adapting their proof to our setting is straightforward.

Proposition 3.3. *Every sentence φ in* BaseGNF *has transitive-closure friendly k-tree-like witnesses, where $k \leq |\varphi|$.*

Hence, it suffices to test satisfiability for BaseGNF restricted to sets of facts with tree decompositions of width $|\varphi| - 1$. It is well known that sets of facts of bounded tree-width can be encoded as trees over a finite alphabet. This makes the satisfiability problem amenable to tree automata techniques, since we can design a tree automaton that runs on representations of these tree decompositions and checks whether some sentence holds in the corresponding set of facts.

Theorem 3.4. *Let φ be a sentence in* BaseGNF *over signature σ, and let $\mathcal{F}_0$ be a finite set of σ-facts. We can construct in* 2EXPTIME *a* 2APT $\mathcal{A}_{\varphi,\mathcal{F}_0}$ *such that*

$$\mathcal{F}_0 \wedge \varphi \textit{ is satisfiable} \quad \textit{iff} \quad L(\mathcal{A}_{\varphi,\mathcal{F}_0}) \neq \emptyset$$

when each $R_i^+ \in \sigma_{\mathcal{D}}$ is interpreted as the transitive closure of $R_i \in \sigma_{\mathcal{B}}$. The number of states and priorities[2] *of $\mathcal{A}_{\varphi,\mathcal{F}_0}$ is at most exponential in $|\varphi| \cdot |\mathcal{F}_0|$.*

We present the details of this construction in the next section. It can be viewed as an extension of work by Calvanese, De Giacomo, and Vardi (2005), and incorporates ideas from automata for guarded logics by, e.g., Grädel and Walukiewicz (1999). It can also be viewed as an optimization of the construction by Benedikt et al. (2016), which we discuss in Section 3.4.

Theorem 3.4 implies that the language of the automaton $\mathcal{A}_{\Sigma \wedge \neg Q,\mathcal{F}_0}$ is empty iff $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ holds. Because 2-way tree automata emptiness is decidable in time polynomial in the overall size and exponential in the number of states and priorities (Vardi, 1998), this yields the 2EXPTIME bound for Theorem 3.1.

3.2 Automata for BaseGNF (Proof of Theorem 3.4)

Fix the signature $\sigma = \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$. As described above in Proposition 3.3, in order to test satisfiability of sentences in BaseGNF over σ, it suffices to consider only sets of facts with tree decompositions of some bounded width. We first describe how to encode such sets of facts using trees over a finite tree signature, and then describe how to construct the automaton to prove Theorem 3.4.

3.2.1 Tree Encodings

Consider a set of σ-facts $\mathcal{F}$, with an $\mathcal{F}_0$-rooted tree decomposition of width $k - 1$ with countable branching, specified by a tree (T, Child) and a function λ. For technical reasons in the automaton construction, it is more convenient to use binary trees, so we want to convert to an alternative tree decomposition of $\mathcal{F}$ based on (T', Child') with a labelling λ' such that (T', Child') is a *full binary tree*, i.e., every node has exactly two children. This can be done by duplicating and rearranging parts

2. With some additional work, it is possible to construct an automaton with only priorities $\{1, 2\}$ (also known as a Büchi automaton), but this optimization is not important for our results and is omitted.

of the tree. First, for each node u, we add infinitely many new children to u, each child being the root of an infinite full binary tree where each node has the same label as u in T. This ensures that each node of T now has infinitely many (but still countably many) children. Second, we convert T into a full binary tree: starting from the root, each node u with children $(v_i)_{i\in\mathbb{N}}$ is replaced by the subtree consisting of $v_1, v_2, \ldots$ and new nodes $u_1, u_2, \ldots$ such that the label at each u_i is the same as the label at u, the left child of u_i is v_i and the right child of u_i is u_{i+1}. In other words, instead of having a node u with infinitely many children $(v_i)_{i\in\mathbb{N}}$, we create an infinite spine of nodes with the same label as u, and attach each v_i to a different copy u_i of u on this spine.

Now we can start encoding this infinite full binary tree decomposition. To achieve this, we specify a finite set U of *names* that can be used to describe the possibly infinite number of elements in $\mathcal{F}$; we will fix the size of this set momentarily. We include $\mathrm{elems}(\mathcal{F}_0)$ in U; these are precisely the names used to describe elements from the initial set of facts $\mathcal{F}_0$. Then we map the elements in $\mathrm{elems}(\mathcal{F}) \setminus \mathrm{elems}(\mathcal{F}_0)$ to a name in U such that the following condition is satisfied: if u and v are neighboring nodes of T, then distinct elements of $\mathrm{elems}(\lambda(u)) \cup \mathrm{elems}(\lambda(v))$ are mapped to distinct names in u and v. Note that every bag either has names $\mathrm{elems}(\mathcal{F}_0)$ or has at most k names. Hence, letting $l := |\mathrm{elems}(\mathcal{F}_0)|$, we know that $2k + l$ possible names suffice to be able to choose different names for distinct elements in neighboring nodes, in a way that does not conflict with the names of elements in $\mathcal{F}_0$. So we choose U to be of size $2k+l$. This assignment of names is encoded using unary relations D_a for each $a \in U$, so that $D_a(v)$ holds iff a is a name that was assigned to an element in v. Facts in $\mathcal{F}$ are encoded using unary relations $R_{\vec{a}}$ for each $R \in \sigma$ of arity n and each n-tuple $\vec{a} \in U^n$, so that $R_{\vec{a}}(v)$ holds iff R holds of the tuple of elements named by $\vec{a}$ at v.

Example 3.5. Figure 3 shows an encoding of the tree decomposition of $\mathcal{F}$ from Example 3.2 (omitting the D_a-relations).

The encodings will sometimes need to specify a valuation for free variables in a formula, so we also introduce relations for this purpose. Recall that we can assume that formulas of width k use some fixed set of k variable names. For each such variable z and each $c \in U$, we introduce a relation $V_{c/z}$; if $V_{c/z}(v)$ holds, then this indicates that the valuation for z is the element named by c at v. We refer to these relations that give a valuation for the free variables as *free variable markers*.

As mentioned in Section 2.6, we also assume that there are unary relations left and right to indicate whether a node is a left or right child of its parent.

This concludes the definition of our encoding scheme. We let $\Sigma^{\text{code}}_{\sigma,k,l}$ denote the *encoding signature* containing the relations described above, and we use the term *$\Sigma^{code}_{\sigma,k,l}$-tree* to refer to an infinite full binary tree over the tree signature $\Sigma^{\text{code}}_{\sigma,k,l}$.

3.2.2 TREE DECODINGS

If a $\Sigma^{\text{code}}_{\sigma,k,l}$-tree satisfies certain consistency properties, then it can be decoded into a set of σ-facts that extends $\mathcal{F}_0$.

Formally, let $\mathrm{names}(v) := \{a \in U : D_a(v) \text{ holds}\}$ be the set of names used for elements in bag v in some tree; we will abuse notation and write $\vec{a} \subseteq \mathrm{names}(v)$ to mean that $\vec{a}$ is a tuple over names from $\mathrm{names}(v)$. Then a *consistent tree T* with respect to $\Sigma^{\text{code}}_{\sigma,k,l}$ and $\mathcal{F}_0$ is a $\Sigma^{\text{code}}_{\sigma,k,l}$-tree such that:

(i) coded facts respect the domain: for all $R_{\vec{a}} \in \Sigma^{\text{code}}_{\sigma,k,l}$ and for all nodes v, if $R_{\vec{a}}(v)$ holds then $\vec{a} \subseteq \mathrm{names}(v)$;

(ii) there is a bijection between the elements and facts represented at the root node and the elements and facts in $\mathcal{F}_0$: $\text{names}(\epsilon) = \text{elems}(\mathcal{F}_0)$; for each fact $R(c_1 \dots c_n) \in \mathcal{F}_0$, the fact $R_{c_1 \dots c_n}(\epsilon)$ holds in the tree; for every $R_{c_1 \dots c_n}(\epsilon)$, the fact $R(z_1 \dots z_n)$ is in $\mathcal{F}_0$;

(iii) if there is a free variable marker for z, then it is unique: for each variable z, there is at most one node v and one name $c \in U$ such that $V_{c/z}(v)$ holds.

Given a consistent tree T, we say nodes u and v are *a-connected* if there is a sequence of nodes $u = w_0, w_1, \dots, w_j = v$ such that w_{i+1} is a neighbor (child or parent) of w_i, and $a \in \text{names}(w_i)$ for all $i \in \{0, \dots, j\}$. We write $[v, a]$ for the equivalence class of a-connected nodes of v. For $\vec{a} = a_1 \dots a_n$, we often abuse notation and write $[v, \vec{a}]$ for the tuple $[v, a_1], \dots, [v, a_n]$.

The *decoding* of T is the set of σ-facts $\mathsf{decode}(T)$ using elements $\{[v, a] : v \in T, a \in \text{names}(v)\}$, where we identify $c \in \text{elems}(\mathcal{F}_0)$ with $[\epsilon, c]$. For each relation $R \in \sigma$ of arity j, we have $R([v_1, a_1], \dots, [v_j, a_j]) \in \mathsf{decode}(T)$ iff there is some $w \in T$ such that $R_{a_1, \dots, a_j}(w)$ holds and $[w, a_i] = [v_i, a_i]$ for all $1 \leq i \leq j$. Note that a single fact $R([v_1, a_1], \dots, [v_j, a_j])$ in $\mathsf{decode}(T)$ might be coded in multiple nodes in T, but there is no requirement that this fact is coded in all nodes that represent $[v_1, a_1], \dots, [v_j, a_j]$.

3.2.3 AUTOMATON CONSTRUCTION

The goal is to build an automaton for a sentence φ in $\mathsf{BaseGNF}$ as needed for Theorem 3.4. An *automaton for a formula* $\psi(x_1, \dots, x_n)$ in $\mathsf{BaseGNF}$ is an automaton $\mathcal{A}_\psi$ such that for all $\Sigma^{\text{code}}_{\sigma,k,l}$-trees T with free variable markers $V_{a_1/x_1}(v_1), \dots, V_{a_n/x_n}(v_n)$, the automaton $\mathcal{A}_\psi$ accepts T iff $\mathsf{decode}(T)$ satisfies $\psi([v_1, a_1], \dots, [v_n, a_n])$ when each $R^+ \in \sigma_{\mathcal{D}}$ is interpreted as the transitive closure of $R \in \sigma_{\mathcal{B}}$. As a warm-up and an example, we start by constructing a $\mathsf{2APT}$ $\mathcal{B}_{R^+(x_1,x_2)}$ for an atomic formula $R^+(x_1, x_2)$ using $R^+ \in \sigma_{\mathcal{D}}$.

Example 3.6. The $\mathsf{2APT}$ $\mathcal{B}_{R^+(x_1,x_2)}$ runs on $\Sigma^{\text{code}}_{\sigma,k,l}$-trees with free variable markers for x_1 and x_2.

We define the state set to be $(U \times \{\textit{next}, \textit{end}\}) \cup \{\textit{start}, \textit{win}, \textit{lose}\}$. The idea is that Eve must navigate to the node v_1 carrying the free variable marker for x_1, find a series of R-facts, and then show that the last element on this guessed R-path corresponds to the element with the free variable marker for x_2. The initial state is *start*, when she is finding the marker for x_1. The states of the form $(a, \textit{next})$ are used to track that a is the name of the furthest element on the R-path that has been found so far. The state $(a, \textit{end})$ is used when she is trying to show that the element with name a is identified by the free variable marker for x_2. If the automaton is in a state of the form $(a, \textit{end})$ or $(a, \textit{next})$ and moves to a node where a is not represented (D_a is not in the label), then she moves to a sink state *lose*. The state *win* is a sink state used when she successfully finds an R-path.

This is implemented by the following transition function, which describes how the automaton should behave in a state from $(U \times \{\textit{next}, \textit{end}\}) \cup \{\textit{start}, \textit{win}, \textit{lose}\}$ and in a node with label β:

$$\delta(\textit{start}, \beta) := \begin{cases} (\mathsf{stay}, (a, \textit{next})) & \text{if } V_{a/x_1} \in \beta \\ (\mathsf{up}, \textit{start}) \vee (\mathsf{left}, \textit{start}) \vee (\mathsf{right}, \textit{start}) & \text{otherwise} \end{cases}$$

$$\delta((a, \textit{next}), \beta) := \begin{cases} (\mathsf{stay}, \textit{lose}) \quad \text{if } D_a \notin \beta & \\ \bigvee\{(\mathsf{stay}, (a', \textit{next})) \vee (\mathsf{stay}, (a', \textit{end})) : R_{a,a'} \in \beta\} & \\ \quad \vee (\mathsf{up}, (a, \textit{next})) \vee (\mathsf{left}, (a, \textit{next})) \vee (\mathsf{right}, (a, \textit{next})) & \text{otherwise} \end{cases}$$

$$\delta((a, \textit{end}), \beta) := \begin{cases} (\mathsf{stay}, \textit{lose}) & \text{if } D_a \notin \beta \\ (\mathsf{stay}, \textit{win}) & \text{if } D_a \in \beta \text{ and } V_{a/x_2} \in \beta \\ (\mathsf{up}, (a, \textit{end})) \vee (\mathsf{left}, (a, \textit{end})) \vee (\mathsf{right}, (a, \textit{end})) & \text{otherwise} \end{cases}$$

$$\delta(\textit{win}, \beta) := (\mathsf{stay}, \textit{win}) \qquad \delta(\textit{lose}, \beta) := (\mathsf{stay}, \textit{lose})$$

Only two priorities are needed. The state *win* is assigned priority 0; all of the other states are assigned priority 1. This prevents Eve from cheating and forever delaying her choice of the elements in the R-path.

As another building block, the next lemma describes how to construct an automaton for a CQ. This lemma is stated for a CQ over signature σ' rather than just σ; the reason for this will become clear when we use this in the inductive case in Lemma 3.9. As usual, we assume that σ' is partitioned into base relations $\sigma'_{\mathcal{B}}$ and distinguished relations $\sigma'_{\mathcal{D}}$.

Lemma 3.7. Given a CQ $\chi(x_1, \ldots, x_j) = \exists \vec{y}(\eta(x_1, \ldots, x_j, \vec{y}))$ of width at most k over signature σ', and given a natural number l, we can construct a $\mathsf{1NPT}$ $\mathcal{N}_\chi$ for the formula χ (over $\Sigma^{\text{code}}_{\sigma',k,l}$-trees).

Furthermore, there is a polynomial function g independent of χ such that the number of states of $\mathcal{N}_\chi$ is at most $2^{g(Kr_\chi)}$ and the number of priorities is at most $g(Kr_\chi)$, where r_χ is the CQ-rank of χ (i.e., the number of conjuncts in η), and $K = 2k + l$. The overall size of the automaton and the running time of the construction is at most exponential in $|\sigma'| \cdot 2^{g(Kr_\chi)}$.

Proof. Each conjunct of η is an atomic formula. For each such atomic formula ψ, we will first describe a $\mathsf{2APT}$ $\mathcal{B}_\psi$ that runs on trees with the free variable markers for $\vec{x}$ and $\vec{y}$ written on the tree. We describe these informally (in terms of choices by Adam and Eve), but they could all be translated into formal automaton definitions in the style of Example 3.6.

- *Base atom.* Suppose ψ is a $\sigma'_{\mathcal{B}}$-atom $A(\vec{z})$, where $\vec{z}$ is a tuple of variables from $\vec{x}$ and $\vec{y}$. Eve tries to navigate to a node v whose label includes fact $A_{\vec{b}}$. If she is able to do this, Adam can then challenge Eve to show that $[v, \vec{b}]$ is the valuation for $\vec{x}$. Say he challenges her on $b_i \in \vec{b}$. Then Eve must navigate from v to the node carrying the marker b_i/x_i. However, she must do this by passing through a series of nodes that also contain b_i. If she is able to do this, $\mathcal{B}_\psi$ enters a sink state with priority 0, so she wins. The other states are assigned priority 1 to force Eve to actually witness $A(\vec{z})$. The number of states of $\mathcal{B}_\psi$ is linear in K, since the automaton must remember the name b_i that Adam is challenging. Only two priorities are needed.

- *Equality.* Suppose ψ is an equality $x_1 = x_2$. Eve navigates to the node v with a free variable marker a/x_1. She is then required to navigate from v to the node carrying the marker for x_2.

She must do so by passing through a series of nodes that also contain a. If she is able to reach the marker a/x_2 in this way, then x_1 and x_2 are marking the same element in the underlying set of facts, so $\mathcal{B}_\psi$ moves to a sink state with priority 0 and she wins. The other states have priority 1, so if Eve is not able to do this, then Adam wins. The state set is of size linear in K, in order to remember the name a. There are two priorities.

- *Distinguished atom.* Suppose ψ is a $\sigma'_{\mathcal{D}}$-atom $R^+(x_1, x_2)$. Then we use the construction in Example 3.6. The number of states in $\mathcal{B}_\psi$ is again linear in K, since it must remember the name a that is currently being processed along this path. There are only two priorities.

This means that we can construct a 2APT for each conjunct in η. We can also easily construct a 2APT that checks that there is a unique free variable marker for each variable in $\vec{y}$; the number of states of this automaton is linear in K, since we must remember which variable and name we are checking. By using closure under intersection of 2APT (see Proposition B.1 in Appendix B), we can construct an automaton $\mathcal{B}_\eta$ that checks that $\eta(\vec{x}, \vec{y})$ holds and that there is a unique free variable marker for each variable in $\vec{y}$. This can be converted to a 1NPT $\mathcal{N}_\eta$ using Theorem 2.5, such that the number of states is exponential in Kr_η, and the number of priorities is polynomial in Kr_η.

Finally, the desired 1NPT $\mathcal{N}_\chi$ for $\chi(\vec{x}) = \exists \vec{y}(\eta(\vec{x}, \vec{y}))$ can be constructed from $\mathcal{N}_\eta$ as follows: it runs on trees with the free variable markers for $\vec{x}$, and simulates $\mathcal{N}_\eta$ while allowing Eve to guess the free variable marker for each variable in $\vec{y}$ (in other words, it is the projection of $\mathcal{N}_\eta$ with respect to the free variable markers for $\vec{y}$; see Proposition B.2 in Appendix B). The idea is that Eve tries to guess a valuation for $\vec{y}$ that satisfies η. Note that for this guessing procedure to be correct, it is essential that $\mathcal{N}_\eta$ is a 1-way nondeterministic automaton, rather than an alternating automaton. The construction of $\mathcal{N}_\chi$ from $\mathcal{N}_\eta$ can be done in polynomial time, with no increase in the number of states or priorities.

Overall, this means that there is some polynomial function g independent of χ such that the number of states of $\mathcal{N}_\chi$ is at most $2^{g(Kr_\chi)}$ and the number of priorities is at most $g(Kr_\chi)$ where r_χ is the CQ-rank of χ, and $K = 2k + l$. It can be checked that the overall size of the automaton and the running time of the construction is at most exponential in $|\sigma'| \cdot 2^{g(Kr_\chi)}$. $\square$

Note that in the base cases of our construction we utilized a simple kind of 2APT with only two priorities. However, in applying Theorem 2.5 we will increase the number of priorities, and thus we are really utilizing the power of 2APT in this construction.

This shows that we can construct an automaton for a CQ over σ, and this could be used to achieve the desired 2EXPTIME bound for satisfiability testing of a CQ. This 2EXPTIME bound can be extended to a UCQ, simply by using closure under union for 1NPT (see Proposition B.1 in Appendix B). However, BaseGNF allows nesting of UCQs with base-guarded negation. The fear is that if we iterate this process for each level of nesting, we will get an exponential blow-up each time, which would lead to non-elementary complexity for satisfiability testing of BaseGNF.

In order to avoid these additional exponential blow-ups, we take advantage of the fact that the nesting of UCQs allowed in BaseGNF is restricted: the free variables in a nested UCQ-shaped formula must be base-guarded, and hence must be represented locally in a single node in the tree code. Recall that a UCQ-shaped formula in BaseGNF with negation depth greater than 0 is of the form $\delta[Y_1 := \alpha_1 \wedge \neg\psi_1, \ldots, Y_s := \alpha_s \wedge \neg\psi_s]$ where δ is a UCQ over the extended signature σ' obtained from σ by adding fresh base relations $Y_1, \ldots, Y_s$, each ψ_i is a UCQ-shaped formula in BaseGNF, and each α_i is a base-guard in σ for the free variables of ψ_i. We first construct an automaton for δ, over the extended signature σ'. The automaton for $\delta[Y_1 := \alpha_1 \wedge \neg\psi_1, \ldots, Y_s := \alpha_s \wedge \neg\psi_s]$ can then simulate the automaton for δ while allowing Eve to guess the valuations for each Y_i (i.e., valuations for each base-guarded subformula $\alpha_i \wedge \neg\psi_i$). In order to prevent Eve from cheating and just guessing that every tuple satisfies these subformulas, Adam is allowed to challenge her on these guesses by launching automata for these subformulas.

The technical difficulty here is that the free variable markers for these subformulas are not written on the tree code any more—they are being guessed on-the-fly by Eve. In order to cope with this, the inductive process will construct an automaton for ψ that can be launched from some internal node v in the tree to test whether or not ψ holds with a local valuation $[v, \vec{a}]$ for $\vec{x}$. The automaton will not specify a single initial state. Instead, there will be a designated initial state for each polarity $p \in \{+, -\}$ and each possible "local assignment" for the free variables $\vec{x}$. A *local assignment* $\vec{a}/\vec{x}$ for $\vec{a} = a_1 \ldots a_n \in U^n$ and $\vec{x} = x_1 \ldots x_n$ is a mapping such that $x_i \mapsto a_i$. A node v in a consistent tree T with $\vec{a} \subseteq \text{names}(v)$ and a local assignment $\vec{a}/\vec{x}$, specifies a valuation for $\vec{x}$. We say it is local since the free variable markers for $\vec{x}$ would all appear locally in v. Given a polarity $p \in \{+, -\}$, we write $p\psi$ for ψ if $p = +$ and $\neg\psi$ if $p = -$.

We will write $\mathcal{A}_\psi$ for the automaton for ψ (without specifying the initial state), and will write $\mathcal{A}_\psi^{p,\vec{a}/\vec{x}}$ for $\mathcal{A}_\psi$ with the designated initial state for p and $\vec{a}/\vec{x}$. We call $\mathcal{A}_\psi$ a *localized automaton for* ψ, since when it is launched from a node v starting from the designated state for p and $\vec{a}/\vec{x}$, it is testing whether $p\psi$ holds when the valuation for $\vec{x}$ is $[v, \vec{a}]$, which is represented locally at v. The point of localized automata is that they can test whether a tuple of elements that appear together in a node satisfy some property, but without having the markers for this tuple explicitly written on the tree. This allows us to "plug-in" localized automata for base-guarded subformulas, as described in the following lemma.

Lemma 3.8. Let η be in $\mathsf{BaseGNF}$ over $\sigma \cup \{Y_1, \ldots, Y_s\}$. Let $\mathcal{A}_\eta$ be a localized $\mathsf{2APT}$ for η over $\Sigma^{\text{code}}_{\sigma\cup\{Y_1,\ldots,Y_s\},k,l}$-trees.

For $1 \leq i \leq s$, let $\chi_i := \alpha_i \wedge \neg\psi_i$ be a formula in $\mathsf{BaseGNF}$ over σ with the number of free variables in χ_i matching the arity of Y_i, and let $\mathcal{A}_{\chi_i}$ be a localized $\mathsf{2APT}$ for χ_i over $\Sigma^{\text{code}}_{\sigma,k,l}$-trees.

We can construct a localized $\mathsf{2APT}$ $\mathcal{A}_\psi$ for $\psi := \eta[Y_1 := \chi_1, \ldots, Y_s := \chi_s]$ over $\Sigma^{\text{code}}_{\sigma,k,l}$-trees in linear time such that the number of states (respectively, priorities) is the sum of the number of states (respectively, priorities) of $\mathcal{A}_\eta, \mathcal{A}_{\chi_1}, \ldots, \mathcal{A}_{\chi_s}$.

We defer the proof of this lemma to Appendix B.3. With the help of Lemma 3.7 and Lemma 3.8, we can now prove our main lemma.

Lemma 3.9. Given a normal form formula $\psi(\vec{x})$ in $\mathsf{BaseGNF}$ of width at most k over signature σ and given a natural number l, we can construct a localized $\mathsf{2APT}$ $\mathcal{A}_\psi$ such that for all consistent $\Sigma^{\text{code}}_{\sigma,k,l}$-trees T, for all polarities $p \in \{+, -\}$, for all local assignments $\vec{a}/\vec{x}$, and for all nodes v in T with $\vec{a} \subseteq \text{names}(v)$,

$$\mathcal{A}_\psi^{p,\vec{a}/\vec{x}} \text{ accepts } T \text{ starting from } v \quad \text{iff} \quad \mathsf{decode}(T) \text{ satisfies } p\psi([v, \vec{a}])$$

when each $R^+ \in \sigma_{\mathcal{D}}$ is interpreted as the transitive closure of $R \in \sigma_{\mathcal{B}}$.

Further, there is a polynomial function f independent of ψ such that the number of states of $\mathcal{A}_\psi$ is at most $N_\psi := f(m_\psi) \cdot 2^{f(Kr_\psi)}$ and the number of priorities is at most $f(Km_\psi)$, where $m_\psi = |\psi|$, r_ψ is the CQ-rank of ψ, and $K = 2k + l$. The overall size of the automaton and the running time of the construction is at most exponential in $|\sigma| \cdot N_\psi$.

Proof. We proceed by induction on the negation depth d of the normal form formula $\psi(\vec{x})$ in BaseGNF. We write m_ψ for $|\psi|$, write r_ψ for the CQ-rank of ψ, and write $N_\psi := f(m_\psi) \cdot 2^{f(Kr_\psi)}$ for some suitably chosen (in particular, non-constant) polynomial f independent of ψ (we will not define f explicitly).

During each case of the inductive construction, we will describe informally how to build the desired automaton, and we will analyze the number of priorities and the number of states required. We defer the analysis of the overall size of the automaton until the end of this proof.

Negation depth 0. For the base case of a UCQ $\psi(\vec{x})$ (negation depth 0), we apply Lemma 3.7 to obtain a 1NPT for each CQ, and then use closure under union to obtain a 1NPT $\mathcal{N}_\psi$ for ψ. This automaton has number of priorities at most polynomial in Km_ψ and number of states at most polynomial in m_ψ and exponential in Kr_ψ, as desired. However, this automaton runs on trees with the free variable markers for $\vec{x}$, so it remains to show that we can construct the automaton $\mathcal{A}_\psi$ required by the lemma, that runs on trees without these markers.

For each local assignment $\vec{a}/\vec{x}$, we can apply the localization theorem (Theorem 2.6) to the set of relation symbols of the form V_{a_i/x_i}, and eliminate the dependence on any other V_{c/x_i} for $c \neq a_i$, by always assuming these relations do not hold. This results in an automaton $\mathcal{A}_\psi^{+,\vec{a}/\vec{x}}$ that no longer relies on free variable markers for $\vec{x}$. By Theorem 2.6, there is only a linear blow-up in the number of states and number of priorities. Then let $\mathcal{A}_\psi^{-,\vec{a}/\vec{x}}$ be the dual of $\mathcal{A}_\psi^{+,\vec{a}/\vec{x}}$, obtained using closure under complement of 2APT (see Proposition B.3 in Appendix B). Finally, we take $\mathcal{A}_\psi$ to be the disjoint union of $\mathcal{A}_\psi^{+,\vec{a}/\vec{x}}$ and $\mathcal{A}_\psi^{-,\vec{a}/\vec{x}}$ over all local assignments $\vec{a}/\vec{x}$; the designated initial state for each $p \in \{+,-\}$ and each localization $\vec{a}/\vec{x}$ is the initial state for $\mathcal{A}_\psi^{p,\vec{a}/\vec{x}}$. All of these automata can be seen to use priorities from the same set, which is of size at most polynomial in Km_ψ. For a suitably chosen f, the number of states in $\mathcal{A}_\psi$ can be bounded by $f(m_\psi) \cdot 2^{f(Kr_\psi)}$ (since there are at most $2K^k$ subautomata being combined) and the number of priorities can be bounded by $f(Km_\psi)$ (since all of the different localized automata $\mathcal{A}_\psi^{p,\vec{a}/\vec{x}}$ being combined use the same set of priorities, which is polynomial in Km_ψ).

Negation depth $d > 0$. The inductive step is for a UCQ-shaped formula ψ with negation depth $d > 0$. Suppose ψ is of the form $\delta[Y_1 := \alpha_1 \wedge \neg\psi_1, \ldots, Y_s := \alpha_s \wedge \neg\psi_s]$ where δ is a UCQ over the extended signature σ' obtained from σ by adding fresh base relations $Y_1, \ldots, Y_s$, each ψ_i is a UCQ-shaped formula in BaseGNF with negation depth strictly less than d and each α_i is a base-guard in σ for the free variables of ψ_i.

Since each α_i and ψ_i have strictly smaller negation depth, the inductive hypothesis yields localized 2APT $\mathcal{A}_{\alpha_i}$ and $\mathcal{A}_{\psi_i}$ for $0 \leq i \leq s$. As a step on the way to constructing an automaton for ψ, we can use these inductively-defined automata to construct a localized 2APT $\mathcal{A}_{\varphi_i}$ for each $\varphi_i := \alpha_i \wedge \neg\psi_i$. Let us fix $0 \leq i \leq s$. We start by taking the disjoint union of $\mathcal{A}_{\alpha_i}$ and $\mathcal{A}_{\psi_i}$. We then add a fresh state $q_0^{p,\vec{a}/\vec{x}}$ to the new automaton for each polarity $p \in \{+,-\}$ and each localization $\vec{a}/\vec{x}$, which becomes the designated initial state for p and $\vec{a}/\vec{x}$ in the new automaton. We keep the transition functions from the subautomata, but add in the following rules for the new states: in state $q_0^{+,\vec{a}/\vec{x}}$, Adam is given the choice of staying in the current node and switching to the designated initial state for $+, \vec{a}/\vec{x}$ in $\mathcal{A}_{\alpha_i}$, or staying in the current node and switching to the designated initial state for $-, \vec{a}/\vec{x}$ in $\mathcal{A}_{\psi_i}$; likewise, in state $q_0^{-,\vec{a}/\vec{x}}$ Eve is given the choice of staying in the current node and switching to the designated initial state for $-, \vec{a}/\vec{x}$ in $\mathcal{A}_{\alpha_i}$ or or staying in the current node and switching to the designated initial state for $+, \vec{a}/\vec{x}$ in $\mathcal{A}_{\psi_i}$. The number of states is at most

$$f(m_{\alpha_i}) \cdot 2^{f(Kr_{\alpha_i})} + f(m_{\psi_i}) \cdot 2^{f(Kr_{\psi_i})} + 2K^k \leq (f(m_{\alpha_i}) + f(m_{\psi_i}) + 1) \cdot 2^{f(Kr_{\alpha_i \wedge \neg\psi_i})}$$

which is at most $N_{\alpha_i \wedge \neg\psi_i}$ by the choice of f. The number of priorities is at most the sum of the number of priorities in $\mathcal{A}_{\alpha_i}$ and $\mathcal{A}_{\psi_i}$, so it is bounded by $f(Km_{\alpha_i \wedge \neg\psi_i})$.

Now let $\mathcal{A}_\delta$ be the 2APT for the underlying UCQ δ obtained as described in the base case. Even though this is over the extended alphabet σ', the overall size of the automaton and the running time of the construction is still at most exponential in $|\sigma| \cdot N_\psi$, since the number of additional relations in σ' is $s \leq |\psi|$. We can then apply Lemma 3.8 to obtain $\mathcal{A}_\psi$ from $\mathcal{A}_\delta, \mathcal{A}_{\varphi_1}, \ldots, \mathcal{A}_{\varphi_s}$. Since the automata $\mathcal{A}_\delta, \mathcal{A}_{\varphi_1}, \ldots, \mathcal{A}_{\varphi_s}$ all satisfy the desired bounds on the number of states, the number of states is at most

$$\begin{aligned} & f(m_\delta) \cdot 2^{f(Kr_\delta)} + f(m_{\varphi_1}) \cdot 2^{f(Kr_{\varphi_1})} + \cdots + f(m_{\varphi_s}) \cdot 2^{f(Kr_{\varphi_s})} \\ \leq\ & (f(m_\delta) + f(m_{\varphi_1}) + \cdots + f(m_{\varphi_s})) \cdot 2^{f(Kr_\psi)} \end{aligned}$$

which is at most N_ψ. Likewise, since the number of priorities is at most the sum of the priorities of $\mathcal{A}_\delta, \mathcal{A}_{\varphi_1}, \ldots, \mathcal{A}_{\varphi_s}$, the number of priorities in $\mathcal{A}_\psi$ can still be bounded by $f(Km_\psi)$.

This concludes the inductive case.

3.2.4 Overall Size

We have argued that each automaton has at most N_ψ states and the number of priorities is polynomial in the size of ψ. It remains to argue that the overall size of $\mathcal{A}_\psi$ is at most exponential in $|\sigma| \cdot N_\psi$. The size of the priority mapping is at most polynomial in N_ψ. The size of the alphabet is exponential in $|\sigma| \cdot K^k$, which is at most exponential in $|\sigma| \cdot N_\psi$. For each state and alphabet symbol, the size of the corresponding transition function formula can always be kept of size at most exponential in N_ψ. Hence, the overall size of the transition function is at most exponential in $|\sigma| \cdot N_\psi$. Together, this means that the overall size of $\mathcal{A}_\psi$ is at most exponential in $|\sigma| \cdot N_\psi$.

It can also be checked that the running time of the construction is polynomial in the size of the constructed automaton, and hence is also exponential in $|\sigma| \cdot N_\psi$. ☐

We must also construct an automaton that checks that the input tree is consistent, and actually represents a set of facts $\mathcal{F}$ such that $\mathcal{F} \supseteq \mathcal{F}_0$ and where every R^+-fact in $\mathcal{F}_0$ is actually witnessed by some path of R-facts in $\mathcal{F}$.

Lemma 3.10. Given a finite set of σ-facts $\mathcal{F}_0$ and natural numbers k and l, we can construct a 2APT $\mathcal{A}_{\mathcal{F}_0}$ in time doubly exponential in $|\sigma| \cdot K$ (for $K = 2k + l$), such that for all $\Sigma^{\text{code}}_{\sigma,k,l}$-trees T,

$$\mathcal{A}_{\mathcal{F}_0} \text{ accepts } T \quad \text{iff} \quad T \text{ is consistent and for all facts } S(\vec{c}) \in \mathcal{F}_0\text{: } \mathsf{decode}(T), [\epsilon, \vec{c}] \text{ satisfies } S(\vec{x})$$

when $R^+ \in \sigma_{\mathcal{D}}$ is interpreted as the transitive closure of $R \in \sigma_{\mathcal{B}}$. The number of states is at most exponential in $|\sigma| \cdot K$, the number of priorities is two, and the overall size is at most doubly exponential in $|\sigma| \cdot K$.

Proof. The automaton is designed to allow Adam to challenge some consistency condition or a particular fact $S(\vec{c})$ in $\mathcal{F}_0$. It is straightforward to design automata for each of the consistency conditions, so we omit the details. To check some base fact $S(\vec{c})$ from $\mathcal{F}_0$, the automaton launches

$\mathcal{A}^{\vec{c}/\vec{x}}_{S(\vec{x})}$ (obtained from Lemma 3.9) from the root. To check some distinguished fact $R^+(c_1, c_2)$, we launch the localized version of $\mathcal{A}_{R^+(x_1,x_2)}$ from Example 3.6 (localized to $c_1/x_1, c_2/x_2$ in the root). Although transitive facts can, in general, mention elements that are far apart in the tree code, we know that c_1, c_2 are both elements of $\mathcal{F}_0$, and hence are both represented locally in the root; hence, it is possible to localize based on this. Note that the R-path witnessing this fact may require elements outside of $\mathrm{elems}(\mathcal{F}_0)$ even though c_1 and c_2 are names of elements in $\mathcal{F}_0$.

It can be checked that the automaton only needs two priorities, the number of states is exponential in $|\sigma| \cdot K$, and the overall size is at most doubly exponential in $|\sigma| \cdot K$. $\square$

3.2.5 CONCLUDING THE PROOF

We can now conclude the proof of Theorem 3.4. We are given some sentence φ in $\mathsf{BaseGNF}$ over signature σ and some finite set of facts $\mathcal{F}_0$. Without loss of generality, we can assume that $|\varphi|\cdot|\mathcal{F}_0| \geq |\sigma|$. We construct the normal form φ' equivalent to φ in exponential time using Proposition 2.4. Although the size of φ' can be exponentially larger than φ, the CQ-rank and width is at most $|\varphi|$. By Proposition 3.3, we know that it suffices to consider only sets of facts with full, binary $\mathcal{F}_0$-rooted tree decompositions of width $|\varphi| - 1$, so we can restrict to considering automata on $\Sigma^{\mathrm{code}}_{\sigma,k,l}$-trees for $k := |\varphi|$ and $l = |\mathrm{elems}(\mathcal{F}_0)|$.

Hence, we apply Lemma 3.9 to φ', k, and l, and construct a $\mathsf{2APT}$ $\mathcal{A}_{\varphi'}$ for φ' (and hence φ) in time exponential in $|\sigma| \cdot N_{\varphi'}$, which is at most doubly exponential in $|\varphi| \cdot |\mathcal{F}_0|$. The number of states and priorities in this automaton is at most singly exponential in $|\varphi| \cdot |\mathcal{F}_0|$.

Next, we apply Lemma 3.10 to $\mathcal{F}_0$, k, and l, to get a $\mathsf{2APT}$ $\mathcal{A}_{\mathcal{F}_0}$. This can be done in time doubly exponential in $|\sigma| \cdot (2k + l)$, which is at most doubly exponential in $|\varphi| \cdot |\mathcal{F}_0|$. The automaton has two priorities and the number of states is at most singly exponential in $|\varphi| \cdot |\mathcal{F}_0|$.

Finally, we construct the desired $\mathsf{2APT}$ $\mathcal{A}_{\varphi,\mathcal{F}_0}$ by taking the disjoint union of the automaton $\mathcal{A}_{\mathcal{F}_0}$ and $\mathcal{A}_{\varphi'}$, and giving Adam an initial choice of which of these automata to simulate. This automaton has a non-empty language iff φ is satisfiable. Moreover, the number of states and priorities in this automaton is still at most singly exponential in $|\varphi| \cdot |\mathcal{F}_0|$. This concludes the proof of Theorem 3.4.

3.3 Consequences for QAtr and Other Variants

We can derive results for QAtr by observing that the QAtc problem subsumes it: to enforce that $R^+ \in \sigma_{\mathcal{D}}$ is transitive, simply interpret it as the transitive closure of a relation R that is never otherwise used. Hence:

Corollary 3.11. *We can decide* $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ *in* $\mathsf{2EXPTIME}$*, where* $\mathcal{F}_0$ *ranges over finite sets of facts,* Σ *over* $\mathsf{BaseGNF}$ *constraints (in particular,* $\mathsf{BaseFGTGD}$*), and* Q *over UCQs.*

In particular, this result holds for *frontier-one TGDs* (those with a single frontier variable), as a single variable is always base-guarded. This answers a question posed in prior work (Baget et al., 2015).

As mentioned in the preliminaries, we have defined QAtr and QAtc based on transitive relations, not reflexive and transitive relations. However, the decidability and combined complexity results described in Theorem 3.1 and Corollary 3.11 (as well as the data complexity results that will be described later in Theorem 3.12 and Theorem 3.13) also apply to the corresponding query answering problems when the distinguished relations are reflexive transitive relations or the reflexive transitive closure of some base relation. Adapting the proofs to this case is a straightforward exercise: the only points that need to be changed are the precise handling of distinguished atoms in Proposition A.1, and the construction of the automaton in Example 3.6 (which is used in Lemmas 3.7 and 3.10).

3.4 Relationship to $\mathsf{GNFP}^{\mathsf{UP}}$

It is well-known that the transitive closure of a binary relation can be expressed in *least fixpoint logic* (LFP), the extension of FO with a least fixpoint operator. LFP can also express that a relation is transitively closed, or is the transitive closure of another relation. Unfortunately, satisfiability is undecidable for FO and hence LFP, so it is not possible to rely on this connection to prove decidability of QAtr or QAtc. On the other hand, the fixpoint extension of GNF (called GNFP) is decidable, but it is unable to express transitive closure (see Bárány et al., 2011; Benedikt et al., 2016), so it also cannot be used to decide QAtr or QAtc.

Recently, a new fixpoint logic called $\mathsf{GNFP}^{\mathsf{UP}}$—*guarded negation fixpoint logic with unguarded parameters*—was introduced (Benedikt et al., 2016). This logic subsumes GNF and GNFP, and is expressive enough to define the transitive closure of a binary relation. It also subsumes a number of other highly expressive Datalog-like languages introduced by Bourhis et al. (2015). However, unlike LFP, satisfiability for $\mathsf{GNFP}^{\mathsf{UP}}$ is decidable (Benedikt et al., 2016). Hence, $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ for $\Sigma \in \mathsf{BaseGNF}$ can be decided by converting $\mathcal{F}_0 \wedge \Sigma \wedge \neg Q$ to an equivalent $\mathsf{GNFP}^{\mathsf{UP}}$ sentence φ, and then testing for unsatisfiability of φ.

Each BaseGNF sentence with distinguished relations R_i^+ can be converted to an equivalent $\mathsf{GNFP}^{\mathsf{UP}}$ sentence with only base relations, since each occurrence of R_i^+ is replaced with a fixpoint formula that describes the transitive closure of the corresponding base relation R_i. The fixpoints in this formula are not nested in complicated ways; using the terminology of Benedikt et al. (2016), they have "parameter-depth" 1. Applying Theorem 20 in Benedikt et al. (2016), this means that QAtc is decidable in 3EXPTIME. Thus, the approach using $\mathsf{GNFP}^{\mathsf{UP}}$ gives an alternative proof of the decidability of query answering with transitivity, but without the optimal 2EXPTIME complexity bound presented here. The automaton construction in this paper can be viewed as an optimization of the automaton construction for $\mathsf{GNFP}^{\mathsf{UP}}$ by Benedikt et al. (2016). The results on $\mathsf{GNFP}^{\mathsf{UP}}$, however, imply that query answering is decidable for BaseGNF not only when we have distinguished R_i^+, but when the distinguished relations are defined by regular expressions over base binary relations and their inverses, in the spirit of C2RPQs (see Example 4 in Benedikt et al., 2016).

Due to the syntactic restrictions in $\mathsf{GNFP}^{\mathsf{UP}}$, the translation described above would not produce a $\mathsf{GNFP}^{\mathsf{UP}}$ formula if distinguished relations were used as guards (i.e. if we started with GNF, rather than BaseGNF). This makes sense, since we will see in Section 6 that our query answering problems become undecidable when the distinguished relations are allowed as guards.

3.5 Data Complexity

Our results in Theorem 3.1 and Corollary 3.11 show upper bounds on the *combined complexity* of the QAtr and QAtc problems. We now turn to the complexity when the query and constraints are fixed but the initial set of facts varies—the *data complexity*.

We first show a CoNP data complexity upper bound for QAtc for BaseGNF constraints.

Theorem 3.12. *For any fixed* BaseGNF *constraints* Σ *and UCQ* Q*, given a finite set of facts* $\mathcal{F}_0$*, we can decide* $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ *in* CoNP *data complexity.*

The algorithm is based on an idea found in earlier CoNP data complexity bounds, used in particular for a guarded variant of fixpoint logic by Bárány, ten Cate, and Otto (2012). From Proposition 3.3, we know that a counterexample to QAtc for any sentence φ and any initial set of facts $\mathcal{F}_0$ can be taken to have an $\mathcal{F}_0$-rooted tree decomposition. While such a decomposition could be large, we can show that to determine whether it satisfies φ it suffices to look at a small amount of information concerning it, in the form of annotations describing, for each $|\varphi|$-tuple $\vec{c}$ in $\mathcal{F}_0$, sufficiently many formulas holding in the subtree that interfaces with $\vec{c}$. We can show a "decomposition theorem" showing that checking φ on a decomposition is equivalent to checking another sentence φ' on the "abstract description of the tentacles" with these annotations. Thus instead of guessing a witness structure, we simply guess a consistent set of annotations and check that φ' holds. We defer the details to Appendix C.2.

For FGTGDs, the data complexity of QA is in PTIME (Baget et al., 2011). We can show that the same holds, but only for BaseCovFGTGDs, and for QAtr rather than QAtc:

Theorem 3.13. *For any fixed* BaseCovFGTGD *constraints* Σ *and base-covered UCQ* Q*, given a finite set of facts* $\mathcal{F}_0$*, we can decide* $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ *in* PTIME *data complexity.*

The proof uses a reduction to the standard QA problem for FGTGDs, and then applies the PTIME result of Baget et al. (2011). The reduction again makes use of tree-likeness to show that we can replace the requirement that the R_i^+ are transitive by the weaker requirement of transitivity within small sets (intuitively, within bags of a decomposition). We will also use this idea for linear orders (see Proposition 4.2), so we defer the proof of this result to Appendix C.3.

As we will see in Section 5, restricting to QAtr in this result is in fact essential to make data complexity tractable, as hardness holds otherwise.

4. Decidability Results for Linear Orders

We now move to QAlin, the setting where the distinguished relations $<_i$ of $\sigma_{\mathcal{D}}$ are *linear* (total) strict orders, i.e., they are transitive, irreflexive, and total.

Unlike the previous section which used base-frontier-guarded constraints, we restrict to *base-covered constraints and queries* in this section. We do this because we will see in Section 6.2 that QAlin is undecidable if we allow base-frontier-guarded constraints, in contrast to the decidability results for QAtc and QAtr with such constraints.

Our main result will again be decidability of QAlin with these additional conditions, but the proof techniques differ from the previous section: instead of using automata, we reduce QAlin to a traditional query answering problem (without distinguished relations), by approximating the linear order axioms in GNF.

4.1 Deciding QAlin by Approximating Linear Order Axioms

We prove the following result about the decidability and combined complexity of QAlin:

Theorem 4.1. *We can decide* $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ *in* 2EXPTIME*, where* $\mathcal{F}_0$ *ranges over finite sets of facts,* Σ *over* BaseCovGNF*, and* Q *over base-covered UCQs. In particular, this holds when* Σ *consists of* BaseCovFGTGD*s.*

Our technique here is to reduce QAlin with BaseCovGNF constraints to traditional QA for GNF constraints. This implies decidability in 2EXPTIME using prior results on QA (Bárány et al., 2012).

The reduction is quite simple, and hence could be applicable to other constraint classes: it simply adds additional constraints that enforce the linear order conditions. However, as we cannot express transitivity or totality in GNF, we only add a weakening of these properties that is expressible in GNF, and then argue that this is sufficient for our purposes. The reduction is described in the following proposition.

Proposition 4.2. *For any finite set of facts* $\mathcal{F}_0$*, constraints* $\Sigma \in \mathsf{BaseCovGNF}$*, and base-covered UCQ* Q*, we can compute* $\mathcal{F}'_0$ *and* $\Sigma' \in \mathsf{BaseGNF}$ *in* PTIME *such that we have* $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ *iff we have* $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q)$*.*

Specifically, $\mathcal{F}'_0$ is $\mathcal{F}_0$ together with facts $G(a, b)$ for every pair $a, b \in \mathrm{elems}(\mathcal{F}_0)$, where G is some fresh binary base relation. We define Σ' as Σ together with the *k-guardedly linear axioms* for each distinguished relation $<$, where k is $\max(|\Sigma \wedge \neg Q|, \mathrm{arity}(\sigma \cup \{G\}))$; namely:

- guardedly total: $\forall xy\ ((\mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y) \wedge \neg(x = y)) \rightarrow x < y \vee y < x)$
- irreflexive: $\neg\exists x\ (x < x)$
- k-guardedly transitive: for $1 \leq l \leq k-1$: $\neg\exists xy\ (\psi_l(x, y) \wedge \mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y) \wedge \neg(x < y))$, and for $1 \leq l \leq k$: $\neg\exists x\ (\psi_l(x, x) \wedge x = x \wedge \neg(x < x))$

where:

- $\mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y)$ is the formula expressing that x, y is guarded by a relation in $\sigma_{\mathcal{B}} \cup \{G\}$ (so it is an existentially-quantified disjunction over all possible atoms using a relation from $\sigma_{\mathcal{B}} \cup \{G\}$ and containing x and y);
- $\psi_1(x, y)$ is just $x < y$; and
- $\psi_l(x, y)$ for $l \geq 2$ is: $\exists x_2 \ldots x_l (x < x_2 \wedge \cdots \wedge x_l < y)$.

Unlike the property of being a linear order, the k-guardedly linear axioms can be expressed in BaseGNF. The idea is that these axioms are strong enough to enforce conditions about transitivity and irreflexivity within "small" sets of elements—intuitively, within sets of at most k elements that appear together in some bag of a $(k-1)$-width tree decomposition.

We now sketch the argument for the correctness of the reduction. The easy direction is where we assume $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q)$ holds, so any $\mathcal{F}' \supseteq \mathcal{F}'_0$ satisfying Σ' must satisfy Q. In this case, consider $\mathcal{F} \supseteq \mathcal{F}_0$ that satisfies Σ and where all $<$ in $\sigma_{\mathcal{D}}$ are strict linear orders. We must show that $\mathcal{F}$ satisfies Q. First, observe that $\mathcal{F}$ satisfies Σ' since the k-guardedly linear axioms for $<$ are clearly satisfied for all k when $<$ is a strict linear order. Now consider the extension of $\mathcal{F}$ to $\mathcal{F}'$ with facts $G(a, b)$ for all $a, b \in \mathrm{elems}(\mathcal{F}_0)$. This must still satisfy Σ': adding these facts means there are additional k-guardedly linear requirements on the elements from $\mathcal{F}_0$, but these requirements already hold since $<$ is a strict linear order. Hence, by our initial assumption, $\mathcal{F}'$ must satisfy Q. Since Q does not mention G, the restriction of $\mathcal{F}'$ back to $\mathcal{F}$ still satisfies Q as well. Therefore, $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ holds.

For the harder direction, we prove the contrapositive of the implication, namely, we suppose that $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q)$ does not hold and show that $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ does not hold either. From our assumption, there is some counterexample $\mathcal{F}' \supseteq \mathcal{F}'_0$ such that $\mathcal{F}'$ satisfies $\Sigma' \wedge \neg Q$. We will again rely on the ability to restrict to tree-like $\mathcal{F}'$, but with a slightly different notion of tree-likeness.

We say a set E of elements from $\mathrm{elems}(\mathcal{F})$ are *base-guarded* in $\mathcal{F}$ if $|E| \leq 1$ or there is some $\sigma_{\mathcal{B}}$-fact or G-fact in $\mathcal{F}$ that mentions all of the elements in E. A *base-guarded-interface tree decomposition* $(T, \mathsf{Child}, \lambda)$ for $\mathcal{F}$ is a tree decomposition satisfying the following additional property: for all nodes n_1 that are not the root of T, if n_2 is a child of n_1 and E is the set of elements mentioned in both n_1 and n_2, then E is base-guarded in $\mathcal{F}$.

Example 4.3. The tree decomposition in Figure 3 is not base-guarded since the set of elements in the interface between the bag with $\{B(c,c), R(b,e)\}$ and the bag with $\{B(c,f), R(e,f), R(f,c)\}$ is $\{c, e\}$, which is not base-guarded in the pictured set $\mathcal{F}$ of facts.

A sentence φ has *base-guarded-interface k-tree-like witnesses* if for any finite set of facts $\mathcal{F}_0$, if there is some $\mathcal{F} \supseteq \mathcal{F}_0$ satisfying φ then there is such an $\mathcal{F}$ with an $\mathcal{F}_0$-rooted $(k-1)$-width base-guarded-interface tree decomposition.

By adapting the proof of tree-like witnesses for GNF in (Bárány et al., 2011) we can show that φ in $\mathsf{BaseGNF}$ have base-guarded-interface k-tree-like witnesses for $k \leq |\varphi|$. This is stated as Proposition A.4 in Appendix A.2. By generalizing this slightly, we can also show the following (see Appendix A.2 again for details of the proof):

Lemma 4.4. The sentence $\Sigma' \wedge \neg Q$ has base-guarded-interface k-tree-like witnesses when taking $k := \max(|\Sigma \wedge \neg Q|, \mathrm{arity}(\sigma \cup \{G\}))$.

Using this lemma, we can assume that we have some counterexample $\mathcal{F}' \supseteq \mathcal{F}_0'$ that satisfies $\Sigma' \wedge \neg Q$ and has a $(k-1)$-width base-guarded-interface tree decomposition. If every $<$ in $\sigma_{\mathcal{D}}$ is a strict linear order in $\mathcal{F}'$, then restricting $\mathcal{F}'$ to the set of σ-facts yields some $\mathcal{F}$ that would satisfy $\Sigma \wedge \neg Q$, i.e., a counterexample allowing us to conclude that $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ does not hold. The problem is that there may be distinguished relations $<$ that are not strict linear orders in $\mathcal{F}'$. In this case, we can show that $\mathcal{F}'$ can be extended to some $\mathcal{F}''$ that still satisfies $\Sigma' \wedge \neg Q$ but where all $<$ in $\sigma_{\mathcal{D}}$ are strict linear orders, which allows us to conclude the proof. Thus, the crucial part of the argument is about extending k-guardedly linear counterexamples to genuine linear orders:

Lemma 4.5. If there is $\mathcal{F}' \supseteq \mathcal{F}_0'$ that has an $\mathcal{F}_0'$-rooted base-guarded-interface $(k-1)$-width tree decomposition and satisfies $\Sigma' \wedge \neg Q$ (and hence is k-guardedly linear), then there is $\mathcal{F}'' \supseteq \mathcal{F}'$ that satisfies $\Sigma' \wedge \neg Q$ where each distinguished relation is a strict linear order.

4.2 Extending Approximate Linear Orders to Genuine Linear Orders (Proof of Lemma 4.5)

The proof of Lemma 4.5 is the main technical result in this section. It require a few auxiliary lemmas (Lemmas 4.6, 4.7, and 4.8 below), that describe the power of the k-guardedly linear axioms in a $(k-1)$-width base-guarded-interface tree decomposition.

Before stating and proving these auxiliary lemmas, we sketch the proof of Lemma 4.5 to give an idea of how these lemmas will be used (see page 219 for more details about the proof). First, sets of facts that have $(k-1)$-width base-guarded-interface tree decompositions and satisfy k-guardedly linear axioms must already be cycle-free with respect to $<$ (this is the Cycles Lemma, Lemma 4.7). Hence, by taking the transitive closure of $<$ in $\mathcal{F}$, we get a new set of facts where every $<$ is a strict *partial* order. Any strict partial order can be further extended to a strict *linear* order using known techniques, so we can obtain $\mathcal{F}'' \supseteq \mathcal{F}'$ where $<$ is a strict linear order. This $\mathcal{F}''$ may have more $<$-facts than $\mathcal{F}'$, but the k-guardedly linear axioms ensure that these new $<$-facts are only about pairs of elements that are not base-guarded (this follows from the Transitivity Lemma, Lemma 4.6).

It is clear that $\mathcal{F}''$ still satisfies the k-guardedly linear axioms, but the fear is that it might not satisfy $\Sigma \wedge \neg Q$. However, this is where use the base-covered assumption on $\Sigma \wedge \neg Q$: satisfiability of $\Sigma \wedge \neg Q$ in $\mathsf{BaseCovGNF}$ is not affected by adding new $<$-facts about pairs of elements that are not base-guarded (this is the Base-coveredness Lemma, Lemma 4.8).

4.2.1 TRANSITIVITY LEMMA

We first prove a result about transitivity for sets of facts with base-guarded-interface tree decompositions.

Lemma 4.6 (Transitivity Lemma)**.** Suppose $\mathcal{F}'$ is a set of facts with an $\mathcal{F}'_0$-rooted $(k-1)$-width base-guarded-interface tree decomposition $(T, \mathsf{Child}, \lambda)$. If $\mathcal{F}'$ is k-guardedly transitive with respect to binary relation $<$, and there is a $<$-path $a_1 \ldots a_n$ where the pair $\{a_1, a_n\}$ is base-guarded, then $a_1 < a_n \in \mathcal{F}'$.

Proof. Suppose there is an $<$-path $a_1 \ldots a_n$ and that the pair $\{a_1, a_n\}$ is base-guarded, with v a node where a_1, a_n appear together. We can assume that $a_1 \ldots a_n$ is a minimal $<$-path between a_1 and a_n, so there are no repeated intermediate elements. Consider a minimal subtree T' of T containing v and containing all of the elements $a_1 \ldots a_n$. We proceed by induction on the length of the path and on the number of nodes of T' (with the lexicographic order on this pair) to show that $a_1 < a_n$ is in $\mathcal{F}'$.

If all elements $a_1 \ldots a_n$ are represented at v, then either (i) all elements are in the root or (ii) the elements are in some internal node. For (i), by construction of $\mathcal{F}'_0$, every pair of elements in $a_1 \ldots a_n$ is guarded (by G). Hence, repeated application of the axiom $\forall xyz((x < z \wedge z < y \wedge \mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y)) \to x < y)$ (which is part of the k-guardedly transitive axioms) is enough to ensure that $a_1 < a_n$ holds. For (ii), since the bag size of an internal node is at most k, we must have $n \leq k$, in which case an application of the k-guardedly transitive axiom to the guarded pair $\{a_1, a_n\}$ ensures that $a_1 < a_n$ holds. This covers the base case of the induction.

Otherwise, there must be some $1 \leq i < j \leq n$ such that a_i and a_j are represented at v, but $a_{i'}$ is not represented at v for $i < i' < j$ (in particular a_{i+1} is not represented at v). We claim that a_i and a_j must be in an interface together.

We say a_{i+1} is *represented in the direction of* v' if v' is a child of v and a_{i+1} is represented in the subtree rooted at v', or v' is the parent of v and a_{i+1} is represented in the tree obtained from T' by removing the subtree rooted at v. Note that by definition of a tree decomposition, since a_{i+1} is not represented at v, it can only be represented in at most one direction.

Let v_{i+1} be the neighbor (child or parent) of v such that a_{i+1} is represented in the direction of v_{i+1}. It is straightforward to show that a_i and a_j must both be represented in the subtree in the direction of v_{i+1} in order to witness the facts $a_i < a_{i+1}$ and $a_{j-1} < a_j$. But a_i and a_j are both in v, so they must both be in v_{i+1}. Hence, a_i and a_j are in the interface between v and v_{i+1}.

If this is an interface with the root node, then the pair a_i, a_j is guarded by G by definition of $\mathcal{F}'_0$. Otherwise, it is base-guarded by definition of base-guarded-interface tree decompositions.

Hence, we can apply the inductive hypothesis to the path $a_i \ldots a_j$ and the subtree T'' of T' in the direction of v_{i+1} to conclude that $a_i < a_j$ holds: the reason why can apply the inductive hypothesis is because T'' is smaller than T' as we removed v, and $a_i \ldots a_j$ is no longer than $a_1 \ldots a_n$. If $i = 1$ and $j = n$, then we are done. If not, then we can apply the inductive hypothesis to the new, strictly shorter path $a_1 \ldots a_i a_j \ldots a_n$ in T' and conclude that $a_1 < a_n$ is in $\mathcal{F}'$ as desired. $\square$

4.2.2 CYCLES LEMMA

We next show that within base-guarded-interface tree decompositions, k-guarded transitivity and irreflexivity imply cycle-freeness.

Lemma 4.7 (Cycles Lemma)**.** Suppose $\mathcal{F}'$ is a set of facts with an $\mathcal{F}_0'$-rooted $(k-1)$-width base-guarded-interface tree decomposition $(T, \mathsf{Child}, \lambda)$. If $\mathcal{F}'$ is k-guardedly transitive and irreflexive with respect to $<$, then $<$ in $\mathcal{F}'$ cannot have a cycle.

Proof. Suppose for the sake of contradiction that there is a cycle $a_1 \ldots a_n a_1$ in $\mathcal{F}'$ using relation $<$. Take a minimal length cycle.

If elements $a_1 \ldots a_n$ are all represented in a single node in T, then either (i) all elements are in the root or (ii) the elements are in some internal node. For (i), by construction of $\mathcal{F}_0'$, every pair of elements in $a_1 \ldots a_n$ is guarded (by G). Hence, repeated application of the axiom $\forall xyz((x < z \wedge z < y \wedge \mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y)) \rightarrow x < y)$ (which is part of the k-guardedly transitive axioms) would force $a_1 < a_1$ to be in $\mathcal{F}'$, which would contradict irreflexivity. Likewise, for (ii), since the bag size of an internal node is at most k, we must have $n \leq k$, so we can apply the k-guardedly transitive axioms to deduce $a_1 < a_1$, which contradicts irreflexivity.

Even if this is not the case, then since $a_n < a_1$ holds, there must be some node v in which both a_1 and a_n are represented. Since not all elements are represented at v, however, there is $1 \leq i < j \leq n$ such that a_i and a_j are represented at v, but $a_{i'}$ is not represented at v for $i < i' < j$. We claim that a_i and a_j must be in an interface together. Observe that a_{i+1} is not represented at v. Let v_{i+1} be the neighbor of v such that a_{i+1} is represented in the subtree in the direction of v_{i+1}. It is straightforward to show that a_i and a_j must both be represented in the subtree of T' in the direction of v_{i+1} in order to witness the facts $a_i < a_{i+1}$ and $a_{j-1} < a_j$. But a_i and a_j are both in v, so they must both be in v_{i+1}. Hence, a_i and a_j are in the interface between v and v_{i+1}. If this is an interface with the root node, then the pair a_i, a_j is base-guarded (by definition of $\mathcal{F}_0'$); otherwise, the definition of base-guarded-interface tree decomposition ensures that they are base-guarded. By the Transitivity Lemma (Lemma 4.6), this means that $a_i < a_j$ holds. Hence, there is a strictly shorter cycle $a_1 \ldots a_i a_j \ldots a_n a_1$, contradicting the minimality of the original cycle. □

4.2.3 BASE-COVEREDNESS LEMMA

Last, we note that adding only facts about unguarded sets of elements cannot impact $\mathsf{BaseCovGNF}$ constraints. This is where we use the base-coveredness assumption.

Lemma 4.8 (Base-coveredness Lemma)**.** Let $\mathcal{F}'' \supseteq \mathcal{F}'$ where $\mathcal{F}''$ contains additional facts about distinguished relations, but no new facts about base-guarded tuples of elements, and where we have $\mathrm{elems}(\mathcal{F}'') = \mathrm{elems}(\mathcal{F}')$. Let $\varphi(\vec{x}) \in \mathsf{BaseCovGNF}$. If $\mathcal{F}', \vec{a}$ satisfies $\varphi(\vec{x})$ then $\mathcal{F}'', \vec{a}$ satisfies $\varphi(\vec{x})$.

Proof sketch. We assume without loss of generality that φ is in normal form $\mathsf{BaseCovGNF}$. Let $\mathsf{BaseCovGNF}^+$ (respectively, $\mathsf{BaseCovGNF}^-$) denote the normal form $\mathsf{BaseGNF}$ formulas where the covering requirements (distinguished atoms in CQ-shaped subformulas are appropriately base-guarded) are required for positively occurring (respectively, negatively occurring) CQ-shaped formulas. Using induction on the negation depth of φ, we can show that:

For $\varphi(\vec{x}) \in \mathsf{BaseCovGNF}^-$: $\mathcal{F}', \vec{a}$ satisfies $\varphi(\vec{x})$ implies $\mathcal{F}'', \vec{a}$ satisfies $\varphi(\vec{x})$.
For $\varphi(\vec{x}) \in \mathsf{BaseCovGNF}^+$: $\mathcal{F}'', \vec{a}$ satisfies $\varphi(\vec{x})$ implies $\mathcal{F}', \vec{a}$ satisfies $\varphi(\vec{x})$.

We omit this straightforward proof. The desired result immediately follows, since $\mathsf{BaseCovGNF} = \mathsf{BaseCovGNF}^-$. □

4.2.4 FINAL PROOF OF LEMMA 4.5

We are now ready to prove Lemma 4.5. We start with some $\mathcal{F}' \subseteq \mathcal{F}'_0$ satisfying $\Sigma' \wedge \neg Q$ with an $\mathcal{F}'_0$-rooted $(k-1)$-width base-guarded-interface tree decomposition. We prove that there is an extension $\mathcal{F}''$ of $\mathcal{F}'$ satisfying $\Sigma' \wedge \neg Q$ in which each distinguished relation is a strict linear order. Note that because $\mathcal{F}'$ satisfies Σ', we know that $\mathcal{F}'$ is k-guardedly linear.

We present the argument when there is one $<$ in $\sigma_{\mathcal{D}}$ that is not a strict linear order in $\mathcal{F}'$, but the argument is similar if there are multiple distinguished relations like this, as we can handle each distinguished relation independently with the method that we will present. Let $\mathcal{G}$ be the extension of $\mathcal{F}'$ obtained by taking $<$ in $\mathcal{G}$ to be the transitive closure of $<$ in $\mathcal{F}'$. Suppose for the sake of contradiction that there is a $<$-cycle in $\mathcal{G}$. We proceed by induction on the number of facts from $\mathcal{G} \setminus \mathcal{F}'$ used in this cycle. If there are no facts from $\mathcal{G} \setminus \mathcal{F}'$ in the cycle, the Cycles Lemma (Lemma 4.7) yields the contradiction. Otherwise, suppose that there is a cycle involving (a_1, a_n), where (a_1, a_n) is a $<$-fact in $\mathcal{G} \setminus \mathcal{F}'$ coming from facts $(a_1, a_2), \ldots, (a_{n-1}, a_n)$ in $\mathcal{F}'$. By replacing (a_1, a_n) in this cycle with $(a_1, a_2), \ldots, (a_{n-1}, a_n)$, we get a (longer) cycle with fewer facts from $\mathcal{G} \setminus \mathcal{F}'$, which is a contradiction by the inductive hypothesis.

Since $<$ is transitive in $\mathcal{G}$ and cycle-free, the relation $<$ in $\mathcal{G}$ must be a strict partial order. We now apply the *order extension principle* or *Szpilrajn extension theorem* (Szpilrajn, 1930): any strict partial order can be extended to a strict total order. From this, we deduce that $\mathcal{G}$ can be further extended by additional $<$-facts to obtain some $\mathcal{F}''$ where $<$ is a strict total order.

We must prove that $\mathcal{F}'' \supseteq \mathcal{G} \supseteq \mathcal{F}' \supseteq \mathcal{F}'_0$ does not include any new $<$-facts about base-guarded tuples. Suppose for the sake of contradiction that there is a new fact $a < b$ in $\mathcal{F}'' \setminus \mathcal{F}'$, where $\{a, b\}$ is base-guarded in $\mathcal{F}'$. By the guardedly total axiom, it must be the case that there was already $b < a$ in $\mathcal{F}'$, and hence also in $\mathcal{F}''$. But $a < b$ and $b < a$ in $\mathcal{F}''$ would together imply $a < a$ in $\mathcal{F}''$, contradicting the fact that $\mathcal{F}''$ is a strict linear order.

Hence, $\mathcal{F}'$ and $\mathcal{F}''$ agree on all facts about base-guarded tuples. Since Q is base-covered and $\Sigma \in \mathsf{BaseCovGNF}$, $\Sigma \wedge \neg Q \in \mathsf{BaseCovGNF}$. Thus, the Base-Coveredness Lemma (Lemma 4.8) guarantees that $\Sigma \wedge \neg Q$ is still satisfied in $\mathcal{F}''$. Since $\mathcal{F}''$ also trivially satisfies all of the k-guardedly linear axioms, it satisfies $\Sigma' \wedge \neg Q$ as required. This concludes the proof of Lemma 4.5.

4.3 Data Complexity

The result of Theorem 4.1 is a combined complexity upper bound. However, as it works by reducing to traditional QA in PTIME, data complexity upper bounds follow from prior work (Bárány et al., 2012).

Corollary 4.9. *For any* $\mathsf{BaseCovGNF}$ *constraints* Σ *and base-covered UCQ* Q*, given a finite set of facts* $\mathcal{F}_0$*, we can decide* $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ *in* CoNP *data complexity.*

This is similar to the way data complexity bounds were shown for QAtr (in Theorem 3.13). However, unlike for the QAtr problem, the constraint rewriting in this section introduces disjunction, so rewriting a QAlin problem for BaseCovFGTGDs does not produce a classical query answering problem for FGTGDs. Thus the rewriting does not imply a PTIME data complexity upper bound for BaseCovFGTGD; indeed, we will see in the next section (in Proposition 5.6) that it is CoNP-hard.

5. Hardness Results

We now show complexity lower bounds. We already know that each one of our variants of QA are 2EXPTIME-hard in combined complexity, and CoNP-hard in data complexity, when GNF constraints are allowed: this follows from existing bounds on GNF reasoning even without distinguished relations (Bárány et al., 2012). However, in some cases, we can show the same hardness results for weaker languages, using the distinguished relations.

In this section, we first summarize our hardness results in Sections 5.1 and 5.2, and then present the proofs in Section 5.3.

5.1 Hardness for QAtc

In the setting where we have distinguished relations interpreted as the transitive closure of other relations, we can show 2EXPTIME-hardness in combined complexity, and CoNP-hardness in data complexity, for the much weaker language of BaseIDs. This is in contrast with Theorem 3.13, which showed PTIME data complexity for QAtr with the more expressive language of BaseCovFGTGDs.

We show hardness via a reduction from QA with *disjunctive inclusion dependencies* (DIDs): recall their definition in Section 2.3. DIDs are known to be 2EXPTIME-hard in combined complexity (Bourhis et al., 2013, Thm. 2) and CoNP-hard in data complexity (Calvanese, Lembo, Lenzerini, & Rosati, 2006; Bourhis et al., 2013), even without distinguished relations. We use transitive closure to emulate disjunction—as was already suggested in the description logic context by Horrocks and Sattler (1999)—by creating an R_i^+-fact and limiting the length of a witness R_i-path using Q'. The choice of the length of the witness path among the possible lengths is used to mimic the disjunction. We thus show:

Theorem 5.1. *For any finite set of facts $\mathcal{F}_0$, DIDs Σ, and UCQ Q on a signature σ, we can compute in PTIME a set of facts $\mathcal{F}_0'$, BaseIDs Σ', and a base-covered CQ Q' on a signature σ' (with a single distinguished relation), such that $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ iff $\mathsf{QAtc}(\mathcal{F}_0', \Sigma', Q')$.*

With the results of Calvanese et al. (2006) and Bourhis et al. (2013), this immediately implies our hardness result:

Corollary 5.2. *The QAtc problem with BaseIDs and base-covered CQs is CoNP-hard in data complexity and 2EXPTIME-hard in combined complexity.*

In fact, the data complexity lower bound for QAtc even holds in the absence of constraints:

Proposition 5.3. *There is a base-covered CQ Q such that $\mathsf{QAtc}(\mathcal{F}_0, \emptyset, Q)$ is CoNP-hard in data complexity.*

We prove this by reducing the problem of 3-coloring a directed graph, known to be NP-hard, to the complement of QAtc: we can easily do this using dependencies with disjunction in the head. Hence, as in the proof of Theorem 5.1, we simulate this disjunction by using a choice of the length of paths that realize transitive closure facts asserted in $\mathcal{F}_0$.

All of these hardness results are first proven using UCQs rather than CQs, and then strengthened by eliminating the disjunction in the query, by adapting a prior trick (see, e.g., Gottlob &

Papadimitriou, 2003) to code the intermediate truth values of disjunctions within a CQ. We state in Appendix E the general lemmas about this transformation, and explain why the proofs of this section still hold when using a CQ rather than a UCQ.

5.2 Hardness for QAlin

Our hardness results for BaseIDs and QAtc also apply to QAlin, using the same technique of translating from DIDs. What changes is the technique used to code disjunction: rather than the length of a path in the transitive closure, we use the totality of the order relation between elements to code disjunction in the relative ordering of elements. We can thus show the following analogue to Theorem 5.1:

Theorem 5.4. *For any finite set of facts $\mathcal{F}_0$, DIDs Σ, and UCQ Q on a signature σ, we can compute in PTIME a set of facts $\mathcal{F}'_0$, BaseIDs Σ' (not mentioning the distinguished relations), and base-covered CQ Q' on a signature σ' (with a single distinguished relation), such that $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ iff $\mathsf{QAlin}(\mathcal{F}'_0, \Sigma', Q')$.*

Hence, we can conclude our hardness result using prior work (Calvanese et al., 2006; Bourhis et al., 2013):

Corollary 5.5. *The QAlin problem with BaseID and base-covered CQs is CoNP-hard in data complexity and 2EXPTIME-hard in combined complexity.*

We can also use a reduction from 3-coloring to show hardness in data complexity even without constraints:

Proposition 5.6. *There is a base-covered CQ Q such that $\mathsf{QAlin}(\mathcal{F}, \emptyset, Q)$ is CoNP-hard in data complexity.*

Again, we will prove the results with UCQs in this section, and explain in Appendix E how to prove these results with a CQ instead.

5.3 Proof of Theorems 5.1 and 5.4

We now start to prove the results of Sections 5.1 and 5.2. In this section, we first prove the results about the translation from DIDs to QAtc and QAlin, namely, Theorems 5.1 and 5.4. In the next section, we show the data complexity hardness results without constraints (Propositions 5.3 and 5.6). We start by proving Theorem 5.1, and we will adapt the proof afterwards to show Theorem 5.4. Recall the claim:

Theorem 5.1. *For any finite set of facts $\mathcal{F}_0$, DIDs Σ, and UCQ Q on a signature σ, we can compute in PTIME a set of facts $\mathcal{F}'_0$, BaseIDs Σ', and a base-covered CQ Q' on a signature σ' (with a single distinguished relation), such that $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ iff $\mathsf{QAtc}(\mathcal{F}'_0, \Sigma', Q')$.*

We will establish a weaker form of the result where Q' is allowed to be a UCQ: the extension where we only use a CQ is shown in Appendix E.4.

5.3.1 DEFINING σ' FROM σ

We create the signature σ' (featuring both base and distinguished relations) from the signature σ of the DIDs and from the DIDs Σ themselves by:

- creating, for each relation R in σ, a base relation R' in σ' whose arity is $\mathrm{arity}(R) + 2$;
- adding a fresh binary base relation E, and taking the transitive closure E^+ of E as the one distinguished relation of σ';
- creating, for each DID τ in Σ written

$$\forall \vec{x}\ R(\vec{x}) \to \bigvee_{1 \leq i \leq n} \exists \vec{y_i}\ R_i(\vec{x}, \vec{y_i}),$$

  a base relation $\mathsf{Witness}_\tau$ in σ' of arity $|\vec{x}| + \sum_i |\vec{y_i}| + 2n + 2$. For simplicity, we will always use the same variables when writing $\mathsf{Witness}_\tau$-atoms, namely, we will write them $\mathsf{Witness}_\tau(\vec{x}, e, f, \vec{y_1}, e_1, f_1, \ldots, \vec{y_n}, e_n, f_n)$.

5.3.2 Defining Σ' from Σ and σ

We then create the $\mathsf{BaseIDs}$ Σ' from the DIDs Σ. First, for each relation R in σ, we create the following BaseID, asserting that the two additional positions of the base relation R' must be connected by an E-path.

$$\tau'_R : \forall \vec{x}\, e\, f\ R'(\vec{x}, e, f) \to E^+(e, f)$$

The intuition is that the failure of the query will impose that this E-path have length at most 2, so it has length either 1 or 2. Facts with a path of length 1 will be called *genuine facts*, which intuitively means that they really hold, and those with a path of length 2 will be called *pseudo-facts*, intuitively meaning that they will be ignored.

Then, for each DID $\tau : \forall \vec{x}\ R(\vec{x}) \to \bigvee_{1 \leq i \leq n} \exists \vec{y_i}\ R_i(\vec{x}, \vec{y_i})$, we create multiple $\mathsf{BaseIDs}$. First, we create a BaseID τ' with a $\mathsf{Witness}_\tau$-fact in the head:

$$\tau' : \forall \vec{x}\, e\, f\, R'(\vec{x}, e, f) \to \exists \vec{y_1}\, e_1\, f_1\ \ldots\ \vec{y_n}\, e_n\, f_n\ \mathsf{Witness}_\tau(\vec{x}, e, f, \vec{y_1}, e_1, f_1, \ldots, \vec{y_n}, e_n, f_n)$$

Then, for $1 \leq i \leq n$, we create the following BaseID τ'_i:

$$\tau'_i : \forall \vec{x}\, e\, f\, \vec{y_1}\, e_1\, f_1\ \ldots\ \vec{y_n}\, e_n\, f_n\ \mathsf{Witness}_\tau(\vec{x}, e, f, \vec{y_1}, e_1, f_1, \ldots, \vec{y_n}, e_n, f_n) \to R'_i(\vec{x}, \vec{y_i}, e_i, f_i)$$

In other words, whenever a DID τ would be applicable on a fact $R'(\vec{c}, e, f)$, we will create a fact $\mathsf{Witness}_\tau(\vec{c}, e, f, \vec{d_1}, e_1, f_1, \ldots, \vec{d_n}, e_n, f_n)$, which will cause *all* head atoms $R'_i(\vec{c}, \vec{d_i}, e_i, f_i)$ for the DID to be instantiated. However, thanks to the two additional positions, we will be free to choose which of these facts are pseudo-facts, and which are genuine. The query will then enforce the correct semantics for DIDs, by prohibiting $\mathsf{Witness}_\tau$-facts whose match was genuine but where all instantiated heads are pseudo-facts.

5.3.3 Defining Q' from Q, σ, and Σ

The UCQ Q' contains the following disjuncts (existentially closed):

- *Q-generated disjuncts*: For each disjunct ψ of the original UCQ Q, we create one disjunct ψ' in the UCQ Q' obtained by replacing each atom $R(\vec{x})$ of ψ by the conjunction $R'(\vec{x}, e, f) \wedge E(e, f)$, where e and f are fresh. That is, the query Q' matches whenever we have a witness for Q consisting of genuine facts.

- *E-path length restriction disjuncts*: For each relation R in σ, we create the following disjunct in Q':
$$R'(\vec{x}, e, f) \wedge E(e, y_1) \wedge E(y_1, y_2) \wedge E(y_2, y_3).$$
This disjunct succeeds if the E-path annotating an R'-fact has length ≥ 3. Hence, for any fact $R'(\vec{a}, e, f)$, the E^+-fact from e to f enforced by the DID τ'_R in Σ must make $R'(\vec{a}, e, f)$ either a genuine fact or a pseudo-fact.

- DID *satisfaction disjuncts*: For every DID $\tau : \forall \vec{x}\ R(\vec{x}) \rightarrow \bigvee_i \exists \vec{y_i}\ R_i(\vec{x}, \vec{y_i})$ in Σ, we create the following disjunct in Q':
$$Q_\tau : \mathsf{Witness}_\tau(\vec{x}, e, f, \vec{y}_1, e_1, f_1, \ldots, \vec{y}_n, e_n, f_n) \wedge E(e, f) \wedge \bigwedge_i \left(E(e_i, w_i) \wedge E(w_i, f_i)\right).$$
Informally, the failure of Q_τ enforces that we cannot have the body of τ holding as a genuine fact and each head disjunct realized by a pseudo-fact.

Observe that all of these disjuncts are trivially base-covered (since they do not use E^+).

5.3.4 Defining $\mathcal{F}'_0$ from $\mathcal{F}_0$

We now explain how to rewrite the facts of an initial fact set $\mathcal{F}_0$ on σ to a fact set $\mathcal{F}'_0$ on σ'. Create $\mathcal{F}'_0$ by replacing each fact $F = R(\vec{a})$ of $\mathcal{F}_0$ by the facts $R'(\vec{a}, b_F, b'_F)$, and $E(b_F, b'_F)$, where b_F and b'_F are fresh. Hence, all facts of $\mathcal{F}_0$ are created as genuine facts.

We have now defined σ', Σ', Q', and $\mathcal{F}'_0$. We now show that the claimed equivalence holds: $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ holds iff $\mathsf{QAtc}(\mathcal{F}'_0, \Sigma', Q')$ holds.

5.3.5 Forward Direction of the Correctness Proof

First, let $\mathcal{F} \supseteq \mathcal{F}_0$ satisfy Σ and violate Q. We must construct $\mathcal{F}'$ that satisfies Σ' and violates Q' when interpreting E^+ as the transitive closure of E.

We construct $\mathcal{F}'$ using the following steps:

- Modify $\mathcal{F}$ in the same way that we used to build $\mathcal{F}'_0$ from $\mathcal{F}_0$ (i.e., expand each fact with two fresh elements with an E-edge between them, to make them genuine facts), yielding $\mathcal{F}_1$. The result of this process consists only of genuine facts, and satisfies all BaseIDs of the form τ'_R.

- Expand $\mathcal{F}_1$ to a superset of facts $\mathcal{F}_2$ by adding facts that solve violations of all dependencies in Σ' of the form τ'.

  Specifically, for every BaseID of the form τ', letting $\tau : \forall \vec{x}\ R(\vec{x}) \rightarrow \bigvee_{1 \leq i \leq n} \exists \vec{y_i}\ S_i(\vec{x}, \vec{y_i})$ be the corresponding DID in Σ, consider a fact $F' = R(\vec{c}, e, f)$ of $\mathcal{F}_1$ that matches the body of τ'. From the way we constructed $\mathcal{F}_1$, we know that it must contain $E(e, f)$, and that $\mathcal{F}$ must contain the fact $F = R(\vec{c})$. Now, as $\mathcal{F}$ satisfies τ, we know that there is $1 \leq i_0 \leq n$ such that $R_{i_0}(\vec{c}, \vec{d}_{i_0})$ holds in $\mathcal{F}$ for some choice of $\vec{d}_{i_0}$. Hence, by construction of $\mathcal{F}_1$ again, we know that it contains $F'_{i_0} = R'_{i_0}(\vec{c}, \vec{d}_{i_0}, e_{i_0}, f_{i_0})$ and $E(e_{i_0}, f_{i_0})$ for some e_{i_0} and f_{i_0}. For every $i \in \{1, \ldots, n\} \backslash \{i_0\}$, create fresh elements $\vec{d}_i, e_i, f_i, w_i$ in the domain of $\mathcal{F}_2$. Now, add to $\mathcal{F}_2$ the fact $F_{\mathrm{w}} = \mathsf{Witness}_\tau(\vec{c}, e, f, \vec{d}_1, e_1, f_1, \ldots, \vec{d}_n, e_n, f_n)$: in this fact, $\vec{c}$ is as in F', $\vec{d}_{i_0}, e_{i_0}, f_{i_0}$ are as in F'_{i_0}, and for the $\vec{d}_i, e_i, f_i$ for $i \neq i_0$ are the fresh elements that we just created.

It is easy to see now that $\mathcal{F}_2$ now satisfies all $\mathsf{BaseIDs}$ of the form τ', and it still satisfies those of the form τ'_R. Further, it is easy to see that for any Witness-fact F_w of $\mathcal{F}_2$ that violates a dependency of the form τ'_i in Σ', the value i must be different from the value i_0 used when creating F_w (as for $i = i_0$ the fact F'_{i_0} considered when creating F_w witnesses that F_w is not a violation of τ_{i_0}). Hence, we have the following property: for any violation of a dependency of Σ' in $\mathcal{F}_2$, the elements to be exported are in $\mathrm{elems}(\mathcal{F}_2)\backslash\mathrm{elems}(\mathcal{F}_1)$, and they only occur in one fact and in one position of $\mathcal{F}_2$.

- We now create $\mathcal{F}_3$ from $\mathcal{F}_2$ by taking care of the remaining violations by performing the *chase* (Abiteboul, Hull, & Vianu, 1995) with Σ' wherever applicable, always creating fresh elements (see Appendix D for details about the chase). Whenever we need to create a witness for some E^+ requirement, we always create an E-path of length 2 with a fresh element in the middle, that is, all facts created in $\mathcal{F}_3\backslash\mathcal{F}_2$ are $\mathsf{Witness}_\tau$-facts and pseudo-facts.

Let $\mathcal{F}' := \mathcal{F}_3$. We now check that $\mathcal{F}'$ is a counterexample to $\mathsf{QAtc}(\mathcal{F}'_0, \Sigma', Q')$. As $\mathcal{F} \supseteq \mathcal{F}_0$, it is clear that $\mathcal{F}_1 \supseteq \mathcal{F}'_0$, so that $\mathcal{F}' \supseteq \mathcal{F}'_0$. Further, it is immediate by definition of the chase that $\mathcal{F}'$ satisfies Σ'. There remains to check that $\mathcal{F}'$ violates Q'. To this end, we will first observe that, by construction of $\mathcal{F}'$, the only E-facts that we create are paths of length 1 on fresh elements in the construction of $\mathcal{F}_1$ from $\mathcal{F}$, and paths of length 2 on fresh elements in the chase in $\mathcal{F}_3$ (these elements were either created as nulls in the chase in $\mathcal{F}_3$, or they were created in $\mathcal{F}_2$ where they only occurred in one fact and at one position). In particular, observe that, whenever we create an E-fact at any point, its endpoints are fresh (they have just been created), so that E-paths have length 1 or 2 and are on pairwise disjoint sets of elements. Hence, as $\mathcal{F}'$ satisfies the τ'_R, any fact $R'(\vec{c}, e, f)$ in $\mathcal{F}'$ is either a genuine fact (i.e., $E(e, f)$ holds in $\mathcal{F}'$) or a pseudo-fact (i.e., there is an E-path of length 2 from e to f in $\mathcal{F}'$), and *these two properties are mutually exclusive*.

We now check that Q' is violated, by considering each possible kind of disjuncts. For the *E-path length restriction disjuncts*, we just explained that the interpretation of E in $\mathcal{F}'$ consists of disjoint paths of length 1 or 2, so there is no E-path of length 3 at all in $\mathcal{F}'$.

For the DID *satisfaction disjuncts*, we will first observe that there are two kinds of $\mathsf{Witness}_\tau$-facts in $\mathcal{F}'$. Some $\mathsf{Witness}_\tau$-facts $\mathsf{Witness}_\tau(\vec{c}, e, f, \vec{d_1}, e_1, f_1, \ldots, \vec{d_n}, e_n, f_n)$ were created in $\mathcal{F}_2$, and for these we always have $E(e, f)$ in $\mathcal{F}_1$ (hence in $\mathcal{F}'$), and the same is true also of $E(e_{i_0}, f_{i_0})$ for the $1 \leq i_0 \leq n$ considered when creating them (using the fact that $\mathcal{F}$ satisfied Σ). All other $\mathsf{Witness}_\tau$-facts of $\mathcal{F}'$ are created in $\mathcal{F}_3$ and include only elements from $\mathrm{elems}(\mathcal{F}_3)\backslash\mathrm{elems}(\mathcal{F}_2)$ or elements occurring only in one position at one fact in $\mathcal{F}_2$ (and not occurring in $\mathcal{F}_1$): hence, for these $\mathsf{Witness}_\tau$-facts, neither $E(e, f)$ holds in $\mathcal{F}'$ nor does $E(e_i, f_i)$ hold for any $1 \leq i \leq n$. This suffices to ensure that disjuncts of the form Q_τ in Q' cannot have a match in $\mathcal{F}'$, because their $\mathsf{Witness}_\tau$-atom can neither match $\mathsf{Witness}_\tau$-facts created in $\mathcal{F}_3$ (as $E(e, f)$ does not hold for them, unlike what Q_τ requires, remembering that paths of length 1 and 2 are mutually exclusive) nor $\mathsf{Witness}_\tau$-facts created in $\mathcal{F}_2$ (because, for $i = i_0$, the fact $E(e_{i_0}, f_{i_0})$ holds for them, violating again what Q_τ requires). Hence, the DID satisfaction disjuncts have no match in $\mathcal{F}'$.

Finally, for the *Q-generated disjuncts*, observe that any match of them must be on genuine facts, i.e., on facts of $\mathcal{F}'$ created for facts of $\mathcal{F}$, so we can conclude because $\mathcal{F}$ violates Q.

Hence, $\mathcal{F}'$ satisfies Σ' and violates Q', which concludes the forward direction.

5.3.6 BACKWARD DIRECTION OF THE CORRECTNESS PROOF

In the other direction, let $\mathcal{F}' \supseteq \mathcal{F}'_0$ be a counterexample to $\mathsf{QAtc}(\mathcal{F}'_0, \Sigma', Q')$. Consider the set of R'-facts from $\mathcal{F}'$ such that $R' \in \sigma'$ corresponds to some $R \in \sigma$ and the elements in the last two positions of this R'-fact are connected by an E-fact, i.e., the genuine facts. Construct a set of facts $\mathcal{F}$ on σ by projecting away the last two positions from these R'-facts, and discarding all of the other facts.

It is clear by construction of $\mathcal{F}'_0$ that $\mathcal{F} \supseteq \mathcal{F}_0$. Further, as $\mathcal{F}'$ violates Q', it is clear that $\mathcal{F}$ violates Q, because any match of a disjunct of Q on $\mathcal{F}$ implies a match of the corresponding Q-generated disjunct Q' in $\mathcal{F}'$. So it suffices to show that $\mathcal{F}$ satisfies Σ.

Hence, assume by contradiction that there is a DID $\tau : \forall \vec{x}\ R(\vec{x}) \rightarrow \bigvee_{1 \leq i \leq n} \exists \vec{y_i}\ S_i(\vec{x}, \vec{y_i})$ of Σ and a fact $R(\vec{c})$ of $\mathcal{F}$ which violates it. Let $F' = R'(\vec{c}, e, f)$ be the fact in $\mathcal{F}'$ from which we created F; we know that $E(e, f)$ holds in $\mathcal{F}'$. Since $\mathcal{F}'$ satisfies τ' in Σ', we know that there are $\vec{d_1}, e_1, f_1, \ldots, \vec{d_n}, e_n, f_n$ such that $\mathsf{Witness}_\tau(\vec{c}, e, f, \vec{d_1}, e_1, f_1, \ldots, \vec{d_n}, e_n, f_n)$ holds. Further, as $\mathcal{F}'$ satisfies the τ'_i for $1 \leq i \leq n$, we know that $S_i(\vec{d_i}, e_i, f_i)$ hold in $\mathcal{F}'$ for all $1 \leq i \leq n$, and as $\mathcal{F}'$ satisfies the τ'_{S_i}, we know that $E^+(e_i, f_i)$ holds, so that there is at least one E-path connecting e_i and f_i. As the E-path length-restriction disjuncts are violated in $\mathcal{F}'$, these E-paths all have length in $\{1, 2\}$, and as the DID satisfaction disjunct Q_τ is violated in $\mathcal{F}'$, there is $1 \leq i_0 \leq n$ such that no path from e_{i_0} to f_{i_0} in $\mathcal{F}$ has length 2, so that some path must have length 1. Hence, $\mathcal{F}$ contains $S_{i_0}(\vec{d_{i_0}}, e_{i_0}, f_{i_0})$ and $E(e_{i_0}, f_{i_0})$, so $\mathcal{F}$ contains $S_{i_0}(\vec{d_{i_0}})$, which witnesses that τ is satisfied on $R(\vec{c})$ in $\mathcal{F}$, a contradiction. Hence, $\mathcal{F}$ satisfies Σ, which concludes the proof of Theorem 5.1.

We now prove Theorem 5.4, which states:

Theorem 5.4. *For any finite set of facts $\mathcal{F}_0$, DIDs Σ, and UCQ Q on a signature σ, we can compute in* PTIME *a set of facts $\mathcal{F}'_0$,* $\mathsf{BaseIDs}$ *Σ' (not mentioning the distinguished relations), and base-covered CQ Q' on a signature σ' (with a single distinguished relation), such that* $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ *iff* $\mathsf{QAlin}(\mathcal{F}'_0, \Sigma', Q')$.

The entire proof is shown by adapting the proof of Theorem 5.1. Again, we show the claim with a base-covered UCQ, and we show the result for a CQ in Appendix E.4

Intuitively, instead of using E^+ to emulate a disjunction on the length of the path to encode genuine facts and pseudo facts, we will use the order relation to emulate disjunction on the same elements: $e < f$ will indicate a genuine fact, whereas $f < e$ will indicate a pseudo-fact, and $e = f$ will be prohibited by the query.

5.3.7 DEFINING σ' FROM σ

The signature σ' is defined as in the proof of Theorem 5.1 except that we do not add the relations E and E^+, but add a relation $<$ as a distinguished relation instead.

5.3.8 DEFINING Σ' FROM Σ AND σ

We also define Σ' as before except that we do not create the $\mathsf{BaseIDs}$ of the form $\forall \vec{x}\, e\, f\ R'(\vec{x}, e, f) \rightarrow E^+(e, f)$. Constraints like this are not necessary because the totality of $<$ already enforces the corresponding property. This means that Σ' does not mention the distinguished relations.

5.3.9 DEFINING Q' FROM Q, σ, AND Σ

The UCQ Q' contains the following disjuncts (existentially closed), which are clearly base-covered:

- *Q-generated disjuncts*: For each disjunct ψ of the original UCQ Q, we create one disjunct ψ' in the UCQ Q' where each atom $R(\vec{x})$ is replaced by the conjunction $R'(\vec{x}, e, f) \wedge e < f$, where e and f are fresh. That is, the query Q' matches whenever we have a witness for Q consisting of genuine facts.

- *Order restriction disjuncts*: For each relation R in σ, we create a disjunct $R'(\vec{x}, e, e)$. Intuitively, failure of this disjunct imposes that, for each relation $R' \in \sigma'$ that stands for a relation $R \in \sigma$, the elements in the two last positions must be different; so every fact must be either a genuine fact or a pseudo-fact.

- DID *satisfaction disjuncts*: For every DID $\tau : \forall \vec{x}\; R(\vec{x}) \rightarrow \bigvee_i \exists \vec{y_i} R_i(\vec{x}, \vec{y_i})$ in Σ, we create the following disjunct in Q':

$$Q_\tau : \mathsf{Witness}_\tau(\vec{x}, e, f, \vec{y}_1, e_1, f_1, \ldots, \vec{y}_n, e_n, f_n) \wedge e < f \wedge \bigwedge_{1 \leq i \leq n} f_i < e_i.$$

  Intuitively, Q_τ is satisfied if the body of τ is matched to a genuine fact but each of the head disjuncts of τ is matched to a pseudo-fact.

5.3.10 DEFINING $\mathcal{F}'_0$ FROM $\mathcal{F}_0$

The process to define $\mathcal{F}'_0$ from $\mathcal{F}_0$ is like in the proof of Theorem 5.1 except that, instead of creating the facts $E(b_F, b'_F)$, we create facts $b_F < b'_F$.

The proof that $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ holds iff $\mathsf{QAlin}(\mathcal{F}'_0, \Sigma', Q')$ holds is similar to the proof for Theorem 5.1, so we sketch the proof and highlight the main differences.

5.3.11 FORWARD DIRECTION OF THE CORRECTNESS PROOF

Let $\mathcal{F} \supseteq \mathcal{F}_0$ satisfy Σ and violate Q, and construct $\mathcal{F}'$ that satisfies Σ' and violates Q' and in which $<$ is an order relation. We do so as follows:

- Build $\mathcal{F}'$ from $\mathcal{F}$ by expanding each fact F with two fresh elements b_F and b'_F and adding the fact $b_F < b'_F$ to make it a genuine fact.

- Create $\mathcal{F}_2$ and $\mathcal{F}_3$ as before, except that pseudo-facts and genuine facts are annotated with $<$-facts rather than E^+-facts.

- Add one step where we construct $\mathcal{F}'$ from $\mathcal{F}_3$ by completing $<$ to be a total order. To do so, observe that $<$ in $\mathcal{F}_3$ must be a partial order, because all order facts that we have created are on disjoint elements (they are of the form $b_F < b'_F$ or $b'_F < b_F$ where b_F and b'_F are elements specific to a fact F). Hence, we define $\mathcal{F}'$ by simply completing $<$ to a total order using the order extension principle (Szpilrajn, 1930). Note that this can never change a genuine fact in a pseudo-fact or vice-versa.

As before it is clear that $\mathcal{F}' \supseteq \mathcal{F}'_0$ and that $\mathcal{F}'$ satisfies Σ' (note that the additional order facts created from $\mathcal{F}_3$ to $\mathcal{F}'$ cannot create a violation of Σ', as it does not mention $<$), and we have made sure that $<$ is a total order. To see why Q' is not satisfied in $\mathcal{F}'$, we proceed exactly as before for the DID satisfaction disjuncts and Q-generated disjuncts, but replacing "having an E-fact between e and f" by "having $e < f$", and replacing "having an E-path of length 2 between e and f" by "having $e > f$", and likewise for e_i and f_i. For the order-restriction disjuncts, we simply observe that for any R'-fact $R'(\vec{a}, e, f)$ in $\mathcal{F}'$, by construction we always have $e \neq f$.

5.3.12 Backward Direction of the Correctness Proof

Suppose we have some counterexample $\mathcal{F}'$ to $\mathsf{QAtc}(\mathcal{F}'_0, \Sigma', Q')$. We construct $\mathcal{F}$ from $\mathcal{F}'$ by keeping all facts whose last two elements e and f are such that $e < f$. The result still clearly satisfies $\mathcal{F} \supseteq \mathcal{F}_0$, and the proof of why it violates Q is unchanged. To show that $\mathcal{F}$ satisfies Σ, we adapt the argument of the proof of Theorem 5.1, but instead of the τ'_{S_i} we rely on totality of the order to deduce that either $e_i < f_i$, $e_i = f_i$, or $f_i < e_i$ for all i, and we rely on the order-restriction disjuncts (rather than the E-path length-restriction disjuncts) to deduce that either $e_i < f_i$ or $f_i < e_i$. We conclude as before by the DID satisfaction disjuncts that we must have $e_i < f_i$ for some i. Thus, we deduce from the satisfaction of Σ' by $\mathcal{F}'$ that $\mathcal{F}$ satisfies Σ, which concludes the backward direction of the correctness proof, and finishes the proof of Theorem 5.4.

5.4 Proof of Propositions 5.3 and 5.6

We now give data complexity lower bounds that show CoNP-hardness even in the absence of constraints. We first prove Proposition 5.3:

Proposition 5.3. *There is a base-covered CQ Q such that $\mathsf{QAtc}(\mathcal{F}_0, \emptyset, Q)$ is CoNP-hard in data complexity.*

Proof. We will show the result for a UCQ Q, and we extend it to a CQ in Appendix E.4. We show CoNP-hardness by reducing the 3-colorability problem in PTIME to the negation of the QAtc problem: this well-known NP-hard problem asks, given an undirected graph $\mathcal{G}$, whether it is 3-colorable, i.e., whether there is a mapping from the vertices of $\mathcal{G}$ to a set of 3 colors (without loss of generality the set $\{1, 2, 3\}$) such that no two adjacent vertices are assigned the same color. Observe that we can modify slightly the definition of this problem to allow vertices to carry multiple colors, i.e., be colored by *non-empty* subsets of $\{1, 2, 3\}$): the use of multiple colors on a vertex imposes more constraints on the vertex, so makes our life harder. In other words, we can restrict the search for solutions to colorings where each vertex has one single color, but when encoding the 3-colorability problem to QAtc we do not need to impose that vertices carry *exactly* one color (we must just impose that they carry *at least* one color).

5.4.1 Definition of the Reduction

We define the signature σ as containing:

- One binary relation G to code the edges of the graph provided as input to the reduction;
- One binary relation E and its transitive closure E^+ (playing a similar role as in the proof of Theorem 5.1);

- For each $\chi \in \{1,2,3\}$, a ternary relation C_χ. Intuitively, the first position of C_χ-facts will contain the element that codes a vertex (and occurs in the G-facts that describe the edges incident to that vertex), and the positions 2 and 3 will contain elements playing a similar role to elements e and f in R'-facts in the proof of Theorem 5.1. Namely, for a fact $C_\chi(a,e,f)$, if e and f are connected by a path of length 1, this will indicate that vertex a has color χ, while if they are connected by a path of length 2 this will indicate that a does not have color χ.

We then define the UCQ Q to contain the following disjuncts (existentially closed):

- *E-path length restriction disjuncts*: For each $\chi \in \{1,2,3\}$, a disjunct that holds if the E-path for C_χ-facts has length ≥ 3:

$$C_\chi(x,e,f) \wedge E(e,y_1) \wedge E(y_1,y_2) \wedge E(y_2,y_3)$$

- *Adjacency disjuncts:* For $\chi \in \{1,2,3\}$, a disjunct Q_i that holds if two adjacent vertices were assigned the same color:

$$C_\chi(x,e,f) \wedge E(e,f) \wedge G(x,x') \wedge C_\chi(x',e',f') \wedge E(e',f')$$

- *Coloring disjunct:* A disjunct that holds if a vertex was not assigned any color:

$$\bigwedge_{\chi \in \{1,2,3\}} C_\chi(x,e_\chi,f_\chi) \wedge E(e_\chi,w_\chi) \wedge E(w_\chi,f_\chi)$$

Given an undirected graph $\mathcal{G}$, we then code it in PTIME as the set of facts $\mathcal{F}_0$ defined by having:

- One fact $G(x,y)$ and one fact $G(y,x)$ for each edge $\{x,y\}$ in $\mathcal{G}$
- One fact $C_\chi(x,e_{x,\chi},f_{x,\chi})$ and one fact $E^+(e_{x,\chi},f_{x,\chi})$ for each vertex x in $\mathcal{G}$ and for each $\chi \in \{1,2,3\}$, where all the $e_{x,\chi}$ and $f_{x,\chi}$ are fresh.

5.4.2 CORRECTNESS PROOF FOR THE REDUCTION

We now show that $\mathcal{G}$ is 3-colorable iff $\mathsf{QAtc}(\mathcal{F}_0,\emptyset,Q)$ is false, completing the reduction.

For the forward direction, consider a 3-coloring of $\mathcal{G}$. Construct $\mathcal{F} \supseteq \mathcal{F}_0$ as follows. For each vertex x of $\mathcal{G}$ (with facts $C_\chi(x,e_{x,\chi},f_{x,\chi}) \in \mathcal{F}_0$ as defined above for all $\chi \in \{1,2,3\}$), create the facts $E(e_{x,\chi},f_{x,\chi})$ where χ is the color assigned to x, and the facts $E(e_{x,\chi'},w_{x,\chi'})$ and $E(w_{x,\chi'},f_{x,\chi'})$ for the two other colors $\chi' \in \{1,2,3\}\backslash\{i\}$ (with the $w_{x,\chi'}$ being fresh). It is clear that $\mathcal{F}$ thus defined is such that $\mathcal{F} \supseteq \mathcal{F}_0$, and that E^+ is the transitive closure of E in $\mathcal{F}$. The E-path length restriction disjuncts of Q do not match in $\mathcal{F}$ (note that we only create E-paths whose endpoints are pairwise distinct), and the coloring disjunct does not match either because each vertex has some color. Finally, the definition of a 3-coloring ensures that the adjacency disjuncts do not match either. Hence, $\mathcal{F}$ is a set of facts violating Q.

For the backward direction, consider some $\mathcal{F} \supseteq \mathcal{F}_0$ where E^+ is the transitive closure of E that violates Q. For any vertex x of $\mathcal{G}$ and for all $\chi \in \{1,2,3\}$, letting $C_\chi(x,e_{x,\chi},f_{x,\chi})$ be the facts as defined in the construction, as the fact $E^+(e_{x,\chi},f_{x,\chi})$ holds in $\mathcal{F}_0$ for each $\chi \in \{1,2,3\}$ and E^+ must be the transitive closure of E in $\mathcal{F}'$, there must be an E-path from $e_{x,\chi}$ to $f_{x,\chi}$. Further, as $\mathcal{F}$

violates the E-path length restriction disjuncts of Q, this path must be of length 1 or 2. Further, as $\mathcal{F}$ violates the coloring disjunct of Q, for every vertex x of $\mathcal{G}$, there must be one $\chi \in \{1, 2, 3\}$ such that the paths for x and χ has length 1. We define a coloring of $\mathcal{G}$ by choosing for each vertex x a color χ for which the E-path has length 1, i.e., the fact $E(e_{x,\chi}, f_{x,\chi})$ holds. (As pointed out in the beginning of the proof, for each x, there could be multiple such χ, but we can take any of them.) This indeed defines a 3-coloring, as any violation of the 3-coloring witnessed by two adjacent vertices of color χ would imply a match of the χ-th adjacency disjunct of Q in $\mathcal{F}$. This concludes the backward direction of the correctness proof of the reduction, and concludes the proof. □

We then modify the proof to show Proposition 5.6:

Proposition 5.6. *There is a base-covered CQ Q such that* $\mathsf{QAlin}(\mathcal{F}, \emptyset, Q)$ *is* CoNP*-hard in data complexity.*

Proof. Again we show the result for a UCQ Q, and extend it to a CQ in Appendix E.4. We define σ as in the proof of Proposition 5.3 but with an order relation $<$ instead of the two relations E and E^+. We define Q as in the proof of Proposition 5.3 but without the E-path length restriction disjunct, and replacing in the other disjuncts the length-1 paths $E(e, f)$ and $E(e', f')$ by $e < f$ and $e' < f'$ and the paths $E(e_\chi, w_\chi) \wedge E(w_\chi, f_\chi)$ by $f_\chi < e_\chi$: the resulting UCQ is clearly base-covered. Note that, unlike in the proof of Theorem 5.4, we need not worry about equalities (so we need not add order restriction disjuncts), as all the relevant elements are already created as distinct elements in $\mathcal{F}_0$. We define $\mathcal{F}_0$ in the same fashion as in the proof of Proposition 5.3 but without the E^+-facts.

We show the same equivalence as in that proof, but for QAlin. We do it by replacing E-paths of length 1 from an e to an f by $e < f$, and E-paths of length 2 by $f < e$. In the forward direction, we build a counterexample set of facts from a coloring as before, and extend $<$ to be an arbitrary total order (this is possible because all $<$-facts that we create are on disjoint pairs). In the backward direction, we use totality of $<$ to argue that a counterexample set of facts must choose some order between the $e_{x,\chi}$ and the $f_{x,\chi}$, and so must decide which colors are assigned to each vertex, in a way that yields a coloring (because the adjacency and coloring disjuncts are violated). □

6. Undecidability Results

We now show how slight changes to the constraint languages and query languages used for the results in Sections 3 and 4 lead to undecidability of query answering.

The undecidability proofs in this section are by reduction from an *infinite tiling problem*, specified by a set of colors $\mathbb{C} = C_1, \ldots, C_k$, a set of forbidden *horizontal patterns* $\mathbb{H} \subseteq \mathbb{C}^2$ and a set of forbidden *vertical patterns* $\mathbb{V} \subseteq \mathbb{C}^2$. It asks, given a sequence $c_0, \ldots, c_n$ of colors of $\mathbb{C}$, whether there exists a function $f : \mathbb{N}^2 \to \mathbb{C}$ such that we have $f((0, i)) = c_i$ for all $0 \leq i \leq n$, and for all $i, j \in \mathbb{N}$, we have $(f(i, j), f(i + 1, j)) \notin \mathbb{H}$ and $(f(i, j), f(i, j + 1)) \notin \mathbb{V}$. It is well-known that there are fixed $\mathbb{C}$, $\mathbb{V}$, $\mathbb{H}$ for which this problem is undecidable (Börger, Grädel, & Gurevich, 1997).

6.1 Undecidability Results for QAtr and QAtc

We have shown in Section 3 that query answering is decidable with transitive relations (even with transitive closure), BaseFGTGDs, and UCQs (Theorem 3.1). Removing the base-frontier-guarded requirement makes QAtc undecidable, even when constraints are inclusion dependencies:

Theorem 6.1. *There is a signature $\sigma = \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$ with a single distinguished relation S^+ in $\sigma_{\mathcal{D}}$, a set Σ of* ID*s on σ, and a CQ Q on $\sigma_{\mathcal{B}}$, such that the following problem is undecidable: given a finite set of facts $\mathcal{F}_0$, decide* $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$.

We can also show that the QAtr problem is undecidable if we allow disjunctive inclusion dependencies which are not base-guarded:

Theorem 6.2. *There is an arity-two signature $\sigma = \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$ with a single distinguished relation S^+ in $\sigma_{\mathcal{D}}$, a set Σ of* DID*s on σ, a CQ Q on $\sigma_{\mathcal{B}}$, such that the following problem is undecidable: given a finite set of facts $\mathcal{F}_0$, decide* $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$.

The two results are incomparable: the second one applies to the QAtr problem rather than QAtc, and does not require a higher-arity signature, but it uses more expressive constraints that feature disjunction. To prove both results, we reduce from a tiling problem, using a transitive successor relation to code a grid, and using the query to test for forbidden adjacent tile patterns. We first present the proof of the second result, because it is simpler. We then adapt this proof to show the first result.

Proof of Theorem 6.2. Fix $\mathbb{C}$, $\mathbb{V}$, $\mathbb{H}$ such that the infinite tiling problem is undecidable. We will give a reduction from this infinite tiling problem to $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$. We prove the result with a UCQ instead of a CQ, and explain how we can use a CQ instead in Appendix E.5.

6.1.1 DEFINITION OF THE REDUCTION

The base relations of the signature are a binary relation S' (for "successor"), one binary relation K_i for each color C_i, and one unary relation K'_i for each color C_i. We also use one distinguished transitive relation, S^+. The idea is that we will create an infinite chain of S' and assert that it is included in S^+: hence, S^+ will be a transitive super-relation of S', so it will contain at least its transitive closure. From S^+, we will define a grid structure on which we can encode the tiling problem, with grid positions represented as a pair of elements in an S^+-fact.

Let Σ consist of the following DIDs (omitting universal quantifiers for brevity):

$$S'(x,y) \to \exists z\, S'(y,z) \qquad S'(x,y) \to S^+(x,y)$$

$$S^+(x,y) \to \bigvee_i K_i(x,y) \qquad S^+(x,y) \to \bigvee_i K_i(y,x) \qquad S^+(x,y) \to \bigvee_i K'_i(x).$$

The $K_i(x,y)$ describe the assignment of colors to grid positions represented as pairs on the infinite chain as we explained. The point of $K'_i(x)$ is that it stands for $K_i(x,x)$: we need a different relation symbol because variable reuse is not allowed in inclusion dependencies. Note that some of these constraints are not base-guarded.

Let the UCQ Q be a disjunction of the following sentences (omitting existential quantifiers for brevity): for each forbidden horizontal pair $(C_i, C_j) \in \mathbb{H}$, with $1 \leq i, j \leq k$, the disjuncts

$$K_i(x,y) \wedge S'(y,y') \wedge K_j(x,y') \quad K'_i(y) \wedge S'(y,y') \wedge K_j(y,y') \quad K_i(y',y) \wedge S'(y,y') \wedge K'_j(y')$$

and for each forbidden vertical pair $(C_i, C_j) \in \mathbb{V}$, the analogous disjuncts

$$K_i(x,y) \wedge S'(x,x') \wedge K_j(x',y) \quad K'_i(x) \wedge S'(x,x') \wedge K_j(x',x) \quad K_i(x,x') \wedge S'(x,x') \wedge K'_j(x').$$

Given an initial instance $c_0, \ldots, c_n$ of the tiling problem, let the initial set of facts $\mathcal{F}_0$ consist of the fact $K'_j(a_0)$ such that C_j is the color of c_0, and for $0 \leq i < n$, the fact $S'(a_i, a_{i+1})$ and the fact $K_j(a_0, a_i)$ such that C_j is the color of initial element c_i.

6.1.2 CORRECTNESS PROOF FOR THE REDUCTION

We claim that the tiling problem has a solution iff there is a superset of $\mathcal{F}_0$ that satisfies Σ and violates Q and where S^+ is transitive. From this we conclude the reduction and deduce the undecidability of QAtr as stated.

For the forward direction, from a solution f to the tiling problem for input $\vec{c}$, we construct the counterexample $\mathcal{F} \supseteq \mathcal{F}_0$ as follows. We first extend the initial chain of S'-facts in $\mathcal{F}_0$ to an infinite chain $S'(a_0, a_1), \ldots, S'(a_m, a_{m+1}), \ldots$, and fix S^+ to be the transitive closure of this S'-chain (so it is indeed transitive). For all $i, j \in \mathbb{N}$ such that $i \neq j$, we create the fact $K_l(a_i, a_j)$ where $l = f(i, j)$. For all $i \in \mathbb{N}$, we create the fact $K'_l(a_i)$ where $l = f(i, i)$. This clearly satisfies the constraints in Σ, and does not satisfy the query because f is a tiling.

For the backward direction, consider an $\mathcal{F} \supseteq \mathcal{F}_0$ that satisfies Σ and violates Q. Starting at the chain of S'-facts of $\mathcal{F}_0$, we can deduce, using the constraints, the existence of an infinite chain $a_0, \ldots, a_n, \ldots$ of S'-facts (whose elements may be distinct or not, this does not matter). Define a tiling f matching the initial tiling problem instance as follows. For all $i < j$ in $\mathbb{N}$, as there is a path of S'-facts from a_i to a_j, we infer that $S^+(a_i, a_j)$ holds, so that $K_l(a_i, a_j)$ holds for some $1 \leq l \leq k$; pick one such fact, taking the fact of $\mathcal{F}_0$ if $i = 0$ and $j \leq n$, and fix $f(i, j) := l$. For $i > j$ we can likewise see that $S^+(a_j, a_i)$ holds whence $K_l(a_i, a_j)$ holds for some l, and we continue as before. For $i \in \mathbb{N}$, as $S'(a_i, a_{i+1})$ holds, we know that $K'_l(a_i)$ holds for some $1 \leq l \leq k$ (again we take the fact of $\mathcal{F}_0$ if $i = 0$), and fix accordingly $f(i, i) := l$. The resulting f clearly satisfies the initial tiling problem instance $c_0, \ldots, c_n$, and it is clearly a solution to the tiling problem, as any forbidden pattern in f would witness a match of a disjunct of Q in $\mathcal{F}$. This shows that the reduction is correct, and concludes the proof. □

We now prove the first result, drawing inspiration from the previous proof, but using the transitive closure to emulate disjunction as we did in Theorem 5.1.

Proof of Theorem 6.1. We reuse the notations for tiling problems from the previous proof. We first prove the result with two distinguished relations S^+ and C^+ and with a UCQ, and then explain how the proof is modified to use only a single transitive relation S^+. The extension to a CQ is explained in Appendix E.5.

6.1.3 DEFINITION OF THE REDUCTION

We define a binary relation S (for "successor") of which S^+ is interpreted as the transitive closure, one binary relation S', one 3-ary relation G (for "grid"), one binary relation G' (standing for cells on the diagonal of the grid, like K'_i in the previous proof), one binary relation T (a terminal for gadgets that we will define to indicate colors) and one binary relation C of which C^+ is interpreted as the transitive closure. The distinction between S and S' is not important for now but will matter when we adapt the proof later to use a single distinguished relation.

We write the following inclusion dependencies Σ (omitting universal quantifiers for brevity):

$$
\begin{array}{lll}
S'(x,y) \to \exists z\, S'(y,z) & S'(x,y) \to S(x,y) & \\
S^+(x,y) \to \exists z\, G(x,y,z) & S^+(x,y) \to \exists z\, G(y,x,z) & S^+(x,y) \to \exists z\, G'(x,z) \\
G(x,y,z) \to \exists w\, T(z,w) & G'(x,z) \to \exists w\, T(z,w) & T(z,w) \to C^+(z,w)
\end{array}
$$

In preparation for defining the query Q, we define $Q_i(z)$ for all $i > 0$ to match the left endpoint of T-facts covered by a C-path of length i (intuitively coding color i):

$$\exists z_1 \dots z_i\, w\; C(z, z_1) \wedge C(z_1, z_2) \wedge \dots, C(z_{i-1}, z_i) \wedge T(z, z_i),$$

The query Q is a disjunction of the following disjuncts (existentially closed):

- *C-path length restriction disjuncts:* One disjunct written as follows, where k is the number of colors

  $$S'(x, w) \wedge G(x, y, z) \wedge\; T(z, z') \wedge C(z, z_1) \wedge\; C(z_1, z_2) \wedge \dots \wedge C(z_{k-1}, z_k), C(z_k, z_{k+1})$$

  and one disjunct defined similarly but with $G(x, y, z)$ replaced by $G'(x, z)$. Intuitively, these disjuncts impose that C-paths annotating T-facts must code colors between 1 and k (i.e., they cannot have length $k + 1$ or greater), and the distinction between G and G' is for reasons similar to the distinction between the K_i and K'_i in the proof of Theorem 6.2.

- *Horizontal adjacency disjuncts:* For each forbidden horizontal pair $(C_i, C_j) \in \mathbb{H}$, with $1 \leq i, j \leq k$, the disjuncts:

  $$G(x, y, z) \wedge G(x, y', z') \wedge Q_i(z) \wedge Q_j(z') \wedge S'(y, y')$$
  $$G'(y, z) \wedge G(y, y', z') \wedge Q_i(z) \wedge Q_j(z') \wedge S'(y, y')$$
  $$G(y', y, z) \wedge G'(y', z') \wedge Q_i(z) \wedge Q_j(z') \wedge S'(y, y')$$

- *Vertical adjacency disjuncts:* For each $(C_i, C_j) \in \mathbb{V}$, the same queries but replacing atoms $S'(y, y')$ by $S'(x, x')$ and the two first atoms of the last two subqueries by $G'(x, z) \wedge G(x, x', z')$ and $G(x, x', z) \wedge G'(x', z')$.

Given an initial instance of the tiling problem $c_0, \dots, c_n$, the initial set of facts $\mathcal{F}_0$ consists of the following: (i) $S'(a_i, a_{i+1})$ for $0 \leq i < n$; (ii) $G(a_0, a_i, b_{0,i})$ for $0 < i \leq n$; (iii) $G'(a_0, b_{0,0})$ (iv) for all $0 \leq i \leq n$, letting l be such that c_i is the l-th color C_l, we create the *length-l gadget on* $b_{0,i}$: we create a path $C(b_{0,i}, d^1_{0,i}), C(d^1_{0,i}, d^2_{0,i}), \dots C(d^{l-1}_{0,i}, d^l_{0,i})$, and the fact $T(b_{0,i}, d^l_{0,i})$, where the elements $b_{0,i}$ and $d^j_{0,i}$ are all fresh;

6.1.4 CORRECTNESS PROOF FOR THE REDUCTION

We claim that the tiling problem has a solution iff there is a superset of $\mathcal{F}_0$ that satisfies Σ and violates Q, where the S^+ and C^+ relations are interpreted as the transitive closure of S and C, from which we conclude the reduction and deduce the undecidability of QAtc as stated.

For the forward direction, from a solution f to the tiling problem for input $\vec{c}$, we construct $\mathcal{F} \supseteq \mathcal{F}_0$ as follows. We first create an infinite chain $S'(a_0, a_1), \dots, S'(a_m, a_{m+1}), \dots$ to complete the initial chain of S'-facts in $\mathcal{F}_0$, we create the implied S-facts, and make S^+ the transitive closure of S. We then create one fact $G(a_i, a_j, b_{i,j})$ for all $i \neq j$ in $\mathbb{N}$ and one fact $G'(a_i, b_{i,i})$ for all $i \in \mathbb{N}$. Last, for all $i, j \in \mathbb{N}$, letting $l := f(i, j)$, we create the length-l gadget on $b_{i,j}$ with fresh elements.

It is clear that $\mathcal{F}$ contains the facts of $\mathcal{F}_0$. It is easy to verify that it satisfies Σ. To see that we do not satisfy the query, observe that:

- The C-path length restriction disjuncts have no match because all C-paths created have length $\leq k$ and are on disjoint sets of elements;

- For the horizontal adjacency disjuncts, it is clear that, in any match, z must be of the form $b_{i,j}$ and z' of the form $b_{i,j+1}$; the reason for the three different forms is that the cases where $i = j$ and where $i \neq j$ are managed differently. Then, as f respects $\mathbb{H}$, we know that the Q_i and Q_j subqueries cannot be satisfied, because for any $l \in \mathbb{N}$ and $i', j' \in \mathbb{N}$, we have $Q_l(b_{i',j'})$ iff $f(i', j') = l$ by construction;
- The reasoning for the vertical adjacency disjuncts is analogous.

Hence, $\mathcal{F} \supseteq \mathcal{F}_0$, satisfies Σ, and violates Q, which concludes the proof of the forward direction of the implication.

For the backward direction, consider an $\mathcal{F} \supseteq \mathcal{F}_0$ that satisfies Σ and violates Q. Starting at the chain of S'-facts of $\mathcal{F}_0$, we can see that there is an infinite chain $a_0, \ldots, a_n, \ldots$ of S'-facts (whose elements may be distinct or not, this does not matter), and hence we infer the existence of the corresponding S-facts. We can also infer the existence of elements $b_{i,j}$ for all $i, j \in \mathbb{N}$ (again, these elements may be distinct or not) such that $G'(a_i, b_{i,i})$ holds and $G(a_i, a_j, b_{i,j})$ holds if $i \neq j$. From this we conclude that there is a fact $T(b_{i,j}, c_{i,j})$ for all $i, j \in \mathbb{N}$, with a C-path from $b_{i,j}$ to $c_{i,j}$. As the C-path length restriction disjuncts are violated, there cannot be such a C-path of length $> k$, so we can define a function f from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{C}$ by setting $f(i, j)$ to be c_l where l is the length of one such path, for all $i, j \in \mathbb{N}$; this can be performed in a way that matches $\mathcal{F}_0$ (by choosing the path that appears in $\mathcal{F}_0$ whenever there is one).

Now, assume by contradiction that f is not a valid tiling. If there are $i, j \in \mathbb{N}$ such that $(f(i, j), f(i, j + 1)) \in \mathbb{H}$, then consider the match $x := a_i$, $y := a_j$, $y' := a_{j+1}$, $z := b_{i,j}$, and $z' := b_{i,j+1}$. If $i \neq j$ and $i \neq j + 1$, we know that $G(a_i, a_j, b_{i,j})$ and $G(a_i, a_{j+1}, b_{i,j+1})$ hold, and taking the witnessing paths used to define $f(i, j)$ and $f(i, j + 1)$, we obtain matches of $Q_{f(i,j)}(b_{i,j})$ and $Q_{f(i,j+1)}(b_{i,j+1})$, so that we obtain a match of one of the disjuncts of Q (one of the first horizontal adjacency disjuncts), a contradiction. The cases where $i = j$ and where $i = j+1$ are similar and correspond to the second and third kinds of horizontal adjacency disjuncts. The case of $\mathbb{V}$ is handled similarly with the vertical adjacency disjuncts. Hence, f is a valid tiling, which concludes the proof of the backward direction of the implication, shows the equivalence, and concludes the reduction and the undecidability proof.

6.1.5 ADAPTING TO A SINGLE DISTINGUISHED RELATION

To prove the result with a single distinguished relation S^+, simply replace all occurrences of C and C^+ in the query and constraints by S and S^+. The rest of the construction is unchanged. The proof of the backwards direction is unchanged, using S in place of C; what must be changed is the proof of the forward direction.

Let f be the solution to the tiling problem. We start by constructing a set of facts $\mathcal{F}_1$ as before from f to complete $\mathcal{F}_0$, replacing the C-facts in the gadgets by S-facts. Now, we complete S^+ to add the transitive closure of these paths (note that they are disjoint from any other S-fact), and complete this to a set of facts to satisfy Σ: create G- and G'-facts, and create gadgets, this time taking all of them to have length $k + 1$: this yields $\mathcal{F}_2$. We repeat this last process indefinitely on the path of S-facts created in the gadgets of the previous iteration, and let $\mathcal{F}$ be the result of this infinite process, which satisfies Σ.

We justify as before that Q has no matches: as we create no S'-facts in $\mathcal{F}_i$ for all $i > 1$, it suffices to observe that no new matches of Q can include any of the new facts, because each disjunct includes an S'-fact. Hence, we can conclude as before. □

6.1.6 RELATED UNDECIDABILITY AND DECIDABILITY RESULTS

The results that we have just shown in Theorem 6.1 and 6.2 complement the undecidability results of Gottlob et al. (2013). Their Theorem 2 shows that QAtr is undecidable for guarded TGDs, two transitive relations and atomic CQs, even with an empty set of initial facts. Their Corollary 1 shows that QAtr is undecidable with guarded disjunctive TGDs (TGDs with disjunction in the head, and with an atom in the body that guards all of the variables in the body) and UCQs, even when restricted to arity-two signatures with a single transitive relation that occurs only in guards, and an empty set of initial facts.

Our results contrast with the decidability results of Baget et al. (2015), which apply to QAtr with linear rules (under a safety condition, which they conjecture is not necessary for decidability): our Theorem 6.1 shows that QAtc with linear rules (without imposing their condition) is undecidable.

6.2 Undecidability Results for QAlin

Section 4 has shown that QAlin is decidable for base-covered CQs and $\mathsf{BaseCovGNF}$ constraints. We now show that dropping the base-covered requirement on the query leads to undecidability:

Theorem 6.3. *There is a signature $\sigma = \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$ where $\sigma_{\mathcal{D}}$ is a single strict linear order relation, a CQ Q on σ, and a set Σ of inclusion dependencies on $\sigma_{\mathcal{B}}$ (i.e., not mentioning the linear order, so in particular base-covered), such that the following problem is undecidable: given a finite set of facts $\mathcal{F}_0$, decide $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$.*

Proof. We show the claim for a UCQ rather than a CQ, and explain in Appendix E.5 how the proof extends to a CQ. As in the proof of Theorem 6.1, we fix an undecidable infinite tiling problem $\mathbb{C}$, $\mathbb{V}$, $\mathbb{H}$, and will reduce that problem to the QAlin problem.

6.2.1 DEFINITION OF THE REDUCTION

We consider the signature consisting of two binary relations R and D (for "right" and "down"), $k-1$ unary relations $K_1, \ldots, K_{k-1}$ (representing the colors), and one unary relation S (representing the fact of being a vertex of the grid — this relation could be rewritten away and is just used to make the inclusion dependencies shorter to write). We also introduce the following abbreviations: (i) we let $K_1'(x)$ stand for $\exists y\ x < y \wedge K_1(y)$; (ii) we let $K_k'(x)$ stand for $\exists y\ x > y \wedge K_{k-1}(y)$; (iii) for all $1 < i < k$, we let $K_i'(x)$ stand for $\exists y y'\ K_{i-1}(y) \wedge y < x \wedge x < y' \wedge K_i(y')$. Intuitively, the K_i' describe the color of elements, which is encoded in their order relation to elements labeled with the K_i.

We put the following inclusion dependencies in Σ:

$$\forall x\, S(x) \to \exists y\, R(x,y) \qquad \forall x\, S(x) \to \exists y\, D(x,y)$$
$$\forall xy\, R(x,y) \to S(y) \qquad \forall xy\, D(x,y) \to S(y)$$

We consider a UCQ formed of the following disjuncts (existentially closed):

$$R(x,y) \wedge D(x,z) \wedge R(z,w) \wedge D(y,w') \wedge w < w'$$
$$R(x,y) \wedge D(x,z) \wedge R(z,w) \wedge D(y,w') \wedge w' < w$$
$$\text{for each } (c,c') \in \mathbb{H} : R(x,y) \wedge K_c'(x) \wedge K_{c'}'(y)$$
$$\text{for each } (c,c') \in \mathbb{V} : D(x,y) \wedge K_c'(x) \wedge K_{c'}'(y)$$

Intuitively, the first two disjuncts enforce a grid structure, by saying that going right and then down must be the same as going down and then right. The other disjuncts enforce that there are no bad horizontal or vertical patterns.

Given an instance $c_0, \dots, c_n$ of the tiling problem, we construct an initial set of facts $\mathcal{F}_0$ consisting of: (i) $S(a_0), \dots, S(a_n)$ for fresh elements $a_0, \dots, a_n$; (ii) $R(a_{i-1}, a_i)$ for $1 \leq i \leq n$ on these elements; (iii) $K_i(b_i)$ for $1 \leq i \leq k$ for fresh elements $b_1, \dots, b_k$; (iv) for each i such that c_i is the color C_1, set $a_i < b_1$ (on the previously defined elements) ; (v) for each i such that c_i is the color C_k, set $a_i > b_{k-1}$ (on the previously defined elements); (vi) for each $1 < j < k$ and i such that c_i is C_j, set $b_{j-1} < a_i$ and $a_i < b_j$ (on the previously defined elements).

6.2.2 CORRECTNESS PROOF FOR THE REDUCTION

Let us show that the reduction is sound. Let us first assume that the tiling problem has a solution f. We construct a counterexample $\mathcal{F} \supseteq \mathcal{F}_0$ as a grid of the R and D relations, with the first elements of the first row being the $a_0, \dots, a_n$, and with the color of elements being coded as their order relations to the b_j like when constructing $\mathcal{F}_0$ above. Complete the interpretation of $<$ to a total order by choosing one arbitrary total order among the elements labeled with the same color, for each color. The resulting interpretation is indeed a total order relation, formed of the following: some total order on the elements of color 1, the element b_1, some total order on the elements of color 2, the element b_2, …, the element b_{k-2}, some total order on the elements of color $k-1$, the element b_{k-1}, some total order on the elements of color k.

It is immediate that the result satisfies Σ. To see why it does not satisfy the first two disjuncts of the UCQ, observe that any match of $R(x, y) \wedge D(x, z) \wedge R(z, w) \wedge D(y, w')$ must have $w = w'$, by construction of the grid in $\mathcal{F}$. To see why it does not satisfy the other disjuncts, notice that any such match must be a pair of two vertical or two horizontal elements; since the elements can match only one K'_c which reflects their assigned color, the absence of matches follows by definition of f being a tiling.

Conversely, let us assume that there exists a counterexample $\mathcal{F} \supseteq \mathcal{F}_0$ which satisfies Σ and violates Q. Clearly, if the first two disjuncts of Q are violated, then, for any element where S holds, considering its R and D successors that exist by Σ, and respectively their D and R successors, we reach the same element. Hence, from $a_0, \dots, a_n$, we can consider the part of $\mathcal{F}$ defined as a grid of the R and D relations, and it is indeed a full grid (R and D edges occur everywhere they should), except that some elements may be reused at multiple places (but this does not matter). Now, we observe that any element except the b_j must be inserted at some position in the total suborder $b_1 < \cdots < b_{k-1}$, so that at least one relation K'_j holds for each element of the grid (several K'_j may hold in case $\mathcal{F}$ has more elements than the b_i that are labeled with the K_i). Choose one of them, in a way that assigns to $a_0, \dots, a_n$ the colors that they had in $\mathcal{F}_0$, and use this to define a function f that extends $a_0, \dots, a_n$. We claim that this f indeed describes a tiling.

Assume by contradiction that it does not. If there are two horizontally adjacent values (i, j) and $(i+1, j)$ realizing a configuration (c, c') from $\mathbb{H}$, by completeness of the grid there is an R-edge between the corresponding elements u, v in $\mathcal{F}$. Further, by the fact that (i, j) and $(i+1, j)$ were given the color that they have in f, we must have $K'_c(u)$ and $K'_c(v)$ in $\mathcal{F}$, so that we must have had a match of a disjunct of Q, a contradiction. The absence of forbidden vertical patterns is proven in the same manner. $\square$

Theorem 6.3 implies that the base-covered requirement is also necessary for constraints:

Corollary 6.4. *There is a signature $\sigma = \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$ where $\sigma_{\mathcal{D}}$ is a single strict linear order relation, and a set Σ' of* $\mathsf{BaseFGTGD}$ *constraints on σ, such that, letting $\top$ be the tautological query, the following problem is undecidable: given a finite set of facts $\mathcal{F}_0$, decide* $\mathsf{QAlin}(\mathcal{F}_0, \Sigma', \top)$.

Proof. To prove Corollary 6.4 from Theorem 6.3, we take constraints Σ' that are equivalent to $\Sigma \wedge \neg Q$, where Σ and Q are as in the previous theorem (in particular, Q is a CQ). Recall that Σ is a set of inclusion dependencies on $\sigma_{\mathcal{B}}$, and therefore are $\mathsf{BaseFGTGDs}$. Hence, it only remains to argue that $\neg Q$ can be written as a $\mathsf{BaseFGTGD}$. Indeed, if $Q = \exists \vec{x}\ \varphi(\vec{x})$ then consider the constraint $\forall \vec{x}(\varphi(\vec{x}) \rightarrow \exists y(y < y))$ where $<$ is the distinguished relation. Since $<$ must be a strict linear order in QAlin, $\exists y(y < y)$ is equivalent to $\bot$ and this new constraint is logically equivalent to $\neg Q$. Moreover, this constraint is trivially in $\mathsf{BaseFGTGD}$ since there are no frontier variables. Hence, $\Sigma \wedge \neg Q$ can be written as a set of $\mathsf{BaseFGTGD}$ constraints as claimed. □

6.2.3 Related Undecidability Results

The result in Theorem 6.3 is related to prior work by Rosati (2007) and Gutiérrez-Basulto et al. (2015), which deals with query answering for UCQs and CQs with inequalities. These results are related because we can transform a query Q using an inequality $x \neq y$ into a new UCQ query $Q' \vee Q''$, where Q' and Q'' is the result of replacing $x \neq y$ in Q with $x < y$ and $y < x$, respectively. Further, we can also express constraints of the form $\forall xy(S_1(x, y) \wedge S_2(x, y) \rightarrow \bot)$, which are part of the description logics considered in those earlier papers, as inclusion dependencies $\forall xy(S_1(x, y) \wedge S_2(x, y) \rightarrow x < x)$. Hence, we could use these prior results to show the undecidability of QAlin for inclusion dependencies and a UCQ over $\sigma = \sigma_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$ when $\sigma_{\mathcal{D}}$ is a single strict linear order. This is weaker than the result stated in Theorem 6.3, which uses a CQ, and in Corollary 6.4, which uses a tautological query.

7. Conclusion

We have given a detailed picture of the impact of transitivity, transitive closure, and linear order restrictions on query answering problems for a broad class of guarded constraints. We have shown that transitive relations and transitive closure restrictions can be handled in guarded constraints as long as the transitive closure relation is not needed as a guard. For linear orders, the same is true if order atoms are covered by base atoms. This implies the analogous results for frontier-guarded TGDs, in particular frontier-one TGDs. We have built upon some known polynomial data complexity upper bounds for classes of guarded constraints without distinguished relations, and have shown how to extend them to the setting of distinguished relations that are required to be transitive or a transitive closure. However, in the case of distinguished relations required to be linear orders, we have shown that PTIME data complexity does not always carry over.

All our results were shown in the absence of constants, so we leave open the question of whether they still hold when constants are allowed in the constraints or queries, though we believe that it should be possible to adapt the proofs. A more important open question is that of deciding entailment over *finite* sets of facts. There are few techniques for deciding entailment over finite sets of facts for logics where it does not coincide with general entailment (and for the constraints considered here it does not coincide). One exception can be found in an earlier work (Kieroński & Tendera, 2007), which establishes decidability for the guarded fragment with transitivity, under the two-variable restriction and assuming that transitive relations appear only in guards. Another exception is in the context of guarded logics (see Bárány and Bojańczyk (2012)), but it is not clear if the techniques there can be extended to our constraint languages.

Acknowledgements

This is an extended version of the conference paper by Amarilli, Benedikt, Bourhis, and Vanden Boom (2016).

Amarilli was partly funded by the Télécom ParisTech Research Chair on Big Data and Market Insights. Bourhis was supported by CPER Nord-Pas de Calais/FEDER DATA Advanced Data Science and Technologies 2015-2020 and ANR Aggreg Project ANR-14-CE25-0017, INRIA Northern European Associate Team Integrated Linked Data. Benedikt was sponsored by the Engineering and Physical Sciences Research Council of the United Kingdom (EPSRC), under grants EP/M005852/1 and EP/L012138/1. Vanden Boom was partially supported by EPSRC grant EP/L012138/1.

Appendix A. Details about Tree Decompositions

Recall the definition of *tree decompositions* from Section 3.1. This appendix presents two kinds of tree decompositions for BaseGNF that were used in the body of the paper. The proofs use a standard technique, involving an unravelling related to a variant of guarded negation bisimulation due to Bárány et al. (2011). A related result and proof also appears in Benedikt et al. (2016).

A.1 Proof of Proposition 3.3: Transitive-closure Friendly Tree Decompositions for BaseGNF

Recall the definition of having *transitive-closure friendly k-tree-like witnesses* from Section 3.1. A sentence φ over σ is said to have transitive-closure friendly k-tree-like witnesses if: for every finite set of $\sigma_{\mathcal{D}}$-facts $\mathcal{F}_0$, if there is a set of facts $\mathcal{F}$ (finite or infinite) extending $\mathcal{F}_0$ with additional $\sigma_{\mathcal{B}}$-facts such that $\mathcal{F}$ satisfies φ when each R^+ is interpreted as the transitive closure of R, then there is such an $\mathcal{F}$ that has an $\mathcal{F}_0$-rooted $(k-1)$-width tree decomposition with countable branching.

In this section, we prove Proposition 3.3:

> **Proposition 3.3.** Every sentence φ in BaseGNF has transitive-closure friendly k-tree-like witnesses, where $k \leq |\varphi|$.

If $|\varphi| < 3$, then φ is necessarily a single 0-ary relation or its negation, in which case the result is trivial, with $k = 1$. Hence, in the rest of the proof, we assume that $|\varphi| \geq 3$, and k will be chosen such that $3 \leq k \leq |\varphi|$; specifically, k will be an upper bound on the maximum number of free variables in any subformula of φ.

A.1.1 BISIMULATION GAME

We say that a set X of elements from $\mathrm{elems}(\mathcal{F})$ is *base-guarded* (or $\sigma_{\mathcal{B}}$-guarded) if $|X| \leq 1$ or there is a $\sigma_{\mathcal{B}}$-fact in $\mathcal{F}$ that uses all of the elements in X. A *partial rigid homomorphism* is a partial homomorphism with respect to all σ-facts in $\mathcal{F}$, such that the restriction to any $\sigma_{\mathcal{B}}$-guarded set of elements is a partial isomorphism.

Let $\mathcal{F}$ and $\mathcal{G}$ be sets of facts extending $\mathcal{F}_0$. The *GN^k bisimulation game* between $\mathcal{F}$ and $\mathcal{G}$ is an infinite game played by two players, Spoiler and Duplicator. The game has two types of positions:

i) partial isomorphisms $f : X \to Y$ or $g : Y \to X$, where $X \subset \mathrm{elems}(\mathcal{F})$ and $Y \subset \mathrm{elems}(\mathcal{G})$ are of size at most k and $\sigma_{\mathcal{B}}$-guarded;

ii) partial rigid homomorphisms $f : X \to Y$ or $g : Y \to X$, where $X \subset \mathrm{elems}(\mathcal{F})$ and $Y \subset \mathrm{elems}(\mathcal{G})$ are of size at most k.

From a type (i) position h, Spoiler must choose a finite subset $X' \subset \mathrm{elems}(\mathcal{F})$ or a finite subset $Y' \subset \mathrm{elems}(\mathcal{G})$, in either case of size at most k, upon which Duplicator must respond with a partial rigid homomorphism h' with domain X' or Y' accordingly. If $h : X \to Y$ and $h' : X' \to Y'$, then h and h' must agree on $X \cap X'$, and if $h : X \to Y$ and $h' : Y' \to X'$, then h^{-1} and h' must agree on $Y \cap Y'$. The analogous property must hold if $h : Y \to X$. In either case, the game then continues from position h'.

From a type (ii) position $h : X \to Y$ (respectively, $h : Y \to X$), Spoiler must choose a finite subset $X' \subset \mathrm{elems}(\mathcal{F})$ (respectively, $Y' \subset \mathrm{elems}(\mathcal{G})$) of size at most k, upon which Duplicator must respond by a partial rigid homomorphism with domain X' (respectively, domain Y'), such that h and h' agree on $X \cap X'$ (respectively, $Y \cap Y'$). Again, the game continues from position h'.

Notice that a type (i) position is a special kind of type (ii) position where Spoiler has the option to *switch the domain* to the other set of facts, rather than just continuing to play in the current domain.

Spoiler wins if he can force the play into a position from which Duplicator cannot respond, and Duplicator wins if she can continue to play indefinitely.

A winning strategy for Duplicator in the GN^k bisimulation game implies agreement between $\mathcal{F}$ and $\mathcal{G}$ on certain BaseGNF formulas.

Proposition A.1. *Let $\mathcal{F}$ and $\mathcal{G}$ be sets of facts extending $\mathcal{F}_0$. Let $\varphi(\vec{x})$ be a formula in* BaseGNF*, and let $k \geq 3$ be greater than or equal to the maximum number of free variables in any subformula of φ.*

If Duplicator has a winning strategy in the GN^k bisimulation game between $\mathcal{F}$ and $\mathcal{G}$ starting from a type (i) or (ii) position $\vec{a} \mapsto \vec{b}$ and $\mathcal{F}$ satisfies $\varphi(\vec{a})$ when interpreting each $R^+ \in \sigma_{\mathcal{D}}$ as the transitive closure of $R \in \sigma_{\mathcal{B}}$, then $\mathcal{G}$ satisfies $\varphi(\vec{b})$ when interpreting each $R^+ \in \sigma_{\mathcal{D}}$ as the transitive closure of $R \in \sigma_{\mathcal{B}}$.

Proof. For this proof, when we talk about sets of facts *satisfying* a formula, we mean satisfaction when interpreting $R^+ \in \sigma_{\mathcal{D}}$ as the transitive closure of $R \in \sigma_{\mathcal{B}}$. We will abuse terminology slightly and say that φ has *width k* if the maximum number of free variables in any subformula of φ is at most k (this is abusing the terminology since we are not assuming in this proof that φ is in normal form).

We proceed by induction on two quantities, ordered lexicographically: first, the number of $\sigma_{\mathcal{D}}$-atoms in φ; second, the size of φ. Suppose Duplicator has a winning strategy in the GN^k bisimulation game between $\mathcal{F}$ and $\mathcal{G}$.

If φ is a $\sigma_{\mathcal{B}}$-atom $A(\vec{x})$, the result follows from the fact that the position $\vec{a} \mapsto \vec{b}$ is a partial homomorphism.

Suppose φ is a $\sigma_{\mathcal{D}}$-atom $R^+(x_1, x_2)$, and $\vec{a} = (a_1, a_2)$ and $\vec{b} = (b_1, b_2)$. If $\mathcal{F}, \vec{a}$ satisfies $R^+(x_1, x_2)$, there is some $n \in \mathbb{N}$ such that $n > 0$ and there is an R-path of length n between a_1 and a_2 in $\mathcal{F}$. We can write a formula $\psi_n(x_1, x_2)$ in BaseGNF (without any $\sigma_{\mathcal{D}}$-atoms) that is satisfied exactly when there is an R-path of length n. Since we do not need to write this in normal form, we can express ψ_n in BaseGNF with width 3 (maximum of 3 free variables in any subformula).

Since $\mathcal{F}, \vec{a}$ satisfies ψ_n and ψ_n does not have any $\sigma_{\mathcal{D}}$-atoms and $k \geq 3$, we can apply the inductive hypothesis from the type (ii) position $\vec{a} \mapsto \vec{b}$ to ensure that $\mathcal{G}, \vec{b}$ satisfies ψ_n, and hence $\mathcal{G}, \vec{b}$ satisfies φ.

If φ is a disjunction, the result follows easily from the inductive hypothesis.

Suppose φ is a base-guarded negation $A(\vec{x}) \wedge \neg\varphi'(\vec{x}')$. By definition of $\mathsf{BaseGNF}$, it must be the case that $A \in \sigma_{\mathcal{B}}$ and $\vec{x}'$ is a sub-tuple of $\vec{x}$. Since $\mathcal{F}, \vec{a}$ satisfies φ, we know that $\mathcal{F}, \vec{a}$ satisfies $A(\vec{x})$, and hence $\vec{a}$ is $\sigma_{\mathcal{B}}$-guarded. This means that $\vec{a} \mapsto \vec{b}$ is actually a partial isomorphism, so we can view it as a position of type (i). This ensures that $\mathcal{G}, \vec{b}$ also satisfies $A(\vec{x})$. It remains to show that it satisfies $\neg\varphi'(\vec{x}')$. Assume for the sake of contradiction that it satisfies $\varphi'(\vec{x}')$. Because $\vec{a} \mapsto \vec{b}$ is a type (i) position, we can consider the move in the game where Spoiler switches the domain to the other set of facts, and then restricts to the elements in the subtuple $\vec{b}'$ of $\vec{b}$ corresponding to $\vec{x}'$ in $\vec{x}$. Let $\vec{a}'$ be the corresponding subtuple of $\vec{a}$. Duplicator must have a winning strategy from the type (i) position $\vec{b}' \mapsto \vec{a}'$, so the inductive hypothesis ensures that $\mathcal{F}, \vec{a}'$ satisfies $\varphi'(\vec{x}')$, a contradiction.

Finally, suppose φ is an existentially quantified formula $\exists y(\varphi'(\vec{x}, y))$. We are assuming that $\mathcal{F}, \vec{a}$ satisfies φ. Hence, there is some $c \in \mathrm{elems}(\mathcal{F})$ such that $\mathcal{F}, \vec{a}, c$ satisfies φ'. Because the width of φ is at most k, we know that the combined number of elements in $\vec{a}$ and c is at most k. Hence, we can consider the move in the game where Spoiler selects the elements in $\vec{a}$ and c. Duplicator must respond with $\vec{b}$ for $\vec{a}$, and some d for c. This is a valid move in the game, so Duplicator must still have a winning strategy from this position $\vec{a}c \mapsto \vec{b}d$, and the inductive hypothesis implies that $\mathcal{G}, \vec{b}, d$ satisfies φ'. Consequently, $\mathcal{G}, \vec{b}$ satisfies φ. $\square$

A.1.2 UNRAVELLING

The tree-like witnesses for Proposition 3.3 can be obtained using an unravelling construction related to the GN^k bisimulation game. This unravelling construction is adapted from Benedikt et al. (2016).

Fix a set of facts $\mathcal{F}$ that extends $\mathcal{F}_0$ with additional $\sigma_{\mathcal{B}}$-facts. Consider the set Π of sequences of the form $X_0 X_1 \ldots X_n$, where $X_0 = \mathrm{elems}(\mathcal{F}_0)$, and for all $i \geq 1$, $X_i \subseteq \mathrm{elems}(\mathcal{F})$ with $|X_i| \leq k$.

We can arrange these sequences in a tree based on the prefix order. Each sequence $\pi = X_0 X_1 \ldots X_n$ identifies a unique node in the tree; we say a is *represented* at node π if $a \in X_n$. For $a \in \mathrm{elems}(\mathcal{F})$, we say π and π' are *a-equivalent* if a is represented at every node on the unique minimal path between π and π' in this tree. For a represented at π, we write $[\pi, a]$ for the a-equivalence class.

The *GN^k-unravelling of* $\mathcal{F}$ is a set of facts $\mathcal{F}^k$ over elements $\{[\pi, a] : \pi \in \Pi \text{ and } a \in \mathrm{elems}(\mathcal{F})\}$ with $S([\pi_1, a_1], \ldots, [\pi_j, a_j]) \in \mathcal{F}^k$ iff $S(a_1, \ldots, a_j) \in \mathcal{F}$ and there is some $\pi \in \Pi$ such that $[\pi, a_i] = [\pi_i, a_i]$ for $i \in \{1, \ldots, j\}$. We can identify $[\epsilon, a]$ with the element $a \in \mathrm{elems}(\mathcal{F}_0)$, so $\mathcal{F}^k$ extends $\mathcal{F}_0$. Hence, there is a natural $\mathcal{F}_0$-rooted tree decomposition of width $k-1$ for $\mathcal{F}^k$ induced by the tree of sequences from Π.

Because this unravelling is related so closely to the GN^k-bisimulation game, it is straightforward to show that Duplicator has a winning strategy in the bisimulation game between $\mathcal{F}$ and its unravelling.

Proposition A.2. *Let $\mathcal{F}$ be a set of facts extending $\mathcal{F}_0$ with additional $\sigma_{\mathcal{B}}$-facts, and let $\mathcal{F}^k$ be the GN^k-unravelling of $\mathcal{F}$. Then Duplicator has a winning strategy in the GN^k bisimulation game between $\mathcal{F}$ and $\mathcal{F}^k$.*

Proof. Given a position f in the GN^k-bisimulation game, we say the *active set* is the set of facts containing the elements in the domain of f. In other words, the active set is either $\mathcal{F}$ or $\mathcal{F}^k$, depending on which set Spoiler is currently playing in. The *safe positions* f in the GN^k-bisimulation game between $\mathcal{F}$ and $\mathcal{F}^k$ are defined as follows: if the active set is $\mathcal{F}^k$, then f is safe if for all $[\pi, a] \in \mathsf{Dom}(f)$, $f([\pi, a]) = a$; if the active set is $\mathcal{F}$, then f is safe if there is some π such that $f(a) = [\pi, a]$ for all $a \in \mathsf{Dom}(f)$.

We now argue that starting from a safe position f, Duplicator has a strategy to move to a new safe position f'. This is enough to conclude that Duplicator has a winning strategy in the GN^k-bisimulation game between $\mathcal{F}$ and $\mathcal{F}^k$ starting from any safe position.

First, assume that the active set is $\mathcal{F}^k$.

- If f is a type (ii) position, then Spoiler can select some new set X' of elements from the active set. Each element in X' is of the form $[\pi', a']$. Duplicator must choose f' such that $[\pi', a']$ is mapped to a' in $\mathcal{F}$, in order to maintain safety. This new position f' is consistent with f on any elements in $X' \cap \mathsf{Dom}(f)$ since f is safe. This f' is still a partial homomorphism since any relation holding for a tuple of elements $[\pi_1, a_1], \ldots, [\pi_n, a_n]$ from $\mathsf{Dom}(f')$ must hold for the tuple of elements $a_1, \ldots, a_n$ in $\mathcal{F}$ by definition of $\mathcal{F}^k$. Consider some element $[\pi', a']$ in $\mathsf{Dom}(f')$. It is possible that there is some $[\pi, a']$ in $\mathsf{Dom}(f')$ with $[\pi, a'] \neq [\pi', a']$; however, $[\pi, a']$ and $[\pi', a']$ are not base-guarded in $\mathcal{F}^k$. Hence, any restriction f'' of f' to a base-guarded set of elements is a bijection. Moreover, such an f'' is a partial isomorphism: consider some $a_1, \ldots, a_n$ in the range of f'' for which some relation S holds in $\mathcal{F}$; since $(f'')^{-1}(a_1), \ldots, (f'')^{-1}(a_n)$ must be base-guarded, we know that there is some π such that $[\pi, a_1] = (f'')^{-1}(a_1)$,..., $[\pi, a_n] = (f'')^{-1}(a_n)$, so by definition of $\mathcal{F}^k$, S holds of $(f'')^{-1}(a_1), \ldots, (f'')^{-1}(a_n)$ as desired. Hence, f' is a safe partial rigid homomorphism.

- If f is a type (i) position, then Spoiler can either choose elements in the active set and we can reason as we did for the type (ii) case, or Spoiler can select elements from the other set of facts.

  We first argue that if Spoiler changes the active set and chooses no new elements, then the game is still in a safe position. Since f is a type (i) position, we know that $\mathsf{Dom}(f)$ is guarded by some base relation S, so there is some π with $f(a) = [\pi, a]$ for all $a \in \mathsf{Dom}(f)$ by construction of $\mathcal{F}^k$. Hence, the new position $f' = f^{-1}$ is still safe.

  If Spoiler switches active sets and chooses new elements, then we can view this as two separate moves: in the first move, Spoiler switches active sets from $\mathcal{F}^k$ to $\mathcal{F}$ but chooses no new elements, and in the second move, Spoiler selects the desired new elements from $\mathcal{F}$. Because switching active sets leads to a safe position (by the argument in the previous paragraph), it remains to define Duplicator's safe strategy when the active set is $\mathcal{F}$, which we explain below.

Now assume that the active set is $\mathcal{F}$. Since f is safe, there is some π such that $f(a) = [\pi, a]$ for all $a \in \mathsf{Dom}(f)$.

- If f is a type (ii) position, then Spoiler can select some new set X' of elements from the active set. We define the new position f' chosen by Duplicator to map each element $a' \in X'$ to $[\pi', a']$ where $\pi' = \pi \cdot X'$. By construction of the unravelling, $\pi' \in \Pi$ and the resulting partial mapping f' still satisfies the safety property with π' as witness. Note that f' is consistent with f for elements of X' that are also in $\mathsf{Dom}(f)$, as we have $[\pi \cdot X', a'] = [\pi, a']$ for $a' \in X' \cap \mathsf{Dom}(f)$. Now consider some tuple $\vec{a} = a_1 \ldots a_n$ of elements from $\mathsf{Dom}(f')$ that are in some relation S. We know that $f'(a_i) = [\pi', a_i]$, hence S must hold for $f'(\vec{a})$ in $\mathcal{F}^k$.

Moreover, for any base-guarded set $\vec{a} = \{a_1, \ldots, a_n\}$ of distinct elements from $\mathsf{Dom}(f')$, $f'(\vec{a})$ must yield a set of distinct elements $\{f'(a_1), \ldots, f'(a_n)\}$, and these elements can only participate in some fact in $\mathcal{F}^k$ if the underlying elements from $\vec{a}$ participate in the same fact in $\mathcal{F}$. Hence, f' is a safe partial rigid homomorphism.

- If f is a type (i) position, then Spoiler can either choose elements in the active set and we can reason as we did for the type (ii) case, or Spoiler can select elements from the other set of facts. It suffices to argue that if Spoiler changes the active set like this, and chooses no new elements, then the game is still in a safe position. But in this case $f' = f^{-1}$ is easily seen to still be safe.

This concludes the proof of Proposition A.2. □

A.1.3 COUNTABLE WITNESSES

The last theorem that we need says that we can always obtain a countable witness to satisfiability of $\mathsf{BaseGNF}$ with transitivity. This follows from known results about least fixed point logic.

Theorem A.3. *For $\varphi \in \mathsf{BaseGNF}$ and a finite set of facts $\mathcal{F}_0$, if there is an $\mathcal{F}$ extending $\mathcal{F}_0$ with $\sigma_\mathcal{B}$-facts and satisfying φ when each relation $R^+ \in \sigma_\mathcal{D}$ is interpreted as the transitive closure of the corresponding $R \in \sigma$, then there is such an $\mathcal{F}$ that has countable cardinality.*

Proof. It is well-known that transitive closure can be expressed in LFP, the extension of FO with a fixpoint operator. Hence, it is possible to rewrite $\varphi \in \mathsf{BaseGNF}$ into $\varphi' \in \mathsf{LFP}$ just over the base relations such that φ is satisfiable (interpreting the distinguished relations as the transitive closure of the corresponding base relations) iff φ' is satisfiable (with no special interpretations). By Theorem 2.1 of (Grädel, 2002), which is essentially a consequence of the Löwenheim-Skolem property, LFP—and hence $\mathsf{BaseGNF}$—has the property that if there is a satisfying set of facts, then there is some satisfying set of facts of countable cardinality. □

A.1.4 CONCLUDING THE PROOF

We can now conclude the proof of Proposition 3.3. Assume that $\mathcal{F}$ is a set of facts extending $\mathcal{F}_0$ with additional $\sigma_\mathcal{B}$-facts such that $\mathcal{F}$ satisfies φ when interpreting R^+ as the transitive closure of R. By Theorem A.3, we can assume that $\mathrm{elems}(\mathcal{F})$ has countable cardinality. Let $3 \leq k \leq |\varphi|$ be an upper bound on the maximum number of free variables in any subformula of φ. It is easy to see that the unravelling $\mathcal{F}^k$ of a countable set of facts $\mathcal{F}$ has countable branching. It remains to show that $\mathcal{F}^k$ satisfies $\mathcal{F}_0$ and φ.

Since $\mathcal{F}$ satisfies φ, Propositions A.1 and A.2 imply that $\mathcal{F}^k$ also satisfies φ when properly interpreting R^+. Moreover, for each $\sigma_\mathcal{D}$-fact $R^+(c,d) \in \mathcal{F}_0$, there is some $n \in \mathbb{N}$ such that $n > 0$ and there is an R-path of length n between c and d in $\mathcal{F}$. We can write a formula $\psi_n(c,d)$ in $\mathsf{BaseGNF}$ (without any $\sigma_\mathcal{D}$-atoms) that is satisfied exactly when there is an R-path of length n. Since ψ_n can be expressed in $\mathsf{BaseGNF}$ with at most 3 free variables in any subformula and $\mathcal{F} \models \psi_n(c,d)$, Propositions A.1 and A.2 imply that $\mathcal{F}^k$ also satisfies ψ_n, and hence $\mathcal{F}^k \models R^+(c,d)$. Thus, we can conclude that the unravelling $\mathcal{F}^k$ is a transitive-closure friendly k-tree-like witness for φ.

A.2 Proof of Lemma 4.4: Base-guarded-interface Tree Decompositions for $\mathsf{BaseGNF}$

Recall the definition of a *base-guarded-interface tree decomposition*, and of having *base-guarded-interface k-tree-like witnesses*, from Section 4.1. In this section, we prove Lemma 4.4 from the main text, with the bulk of the work being to prove that $\mathsf{BaseGNF}$ sentences have base-guarded-interface k-tree-like witnesses.

For the application that we have in mind, we will prove our results for a slight generalization of normal form $\mathsf{BaseGNF}$ formulas that allows $\mathsf{guarded}_{\sigma_{\mathcal{B}}}(x, y)$ (a disjunction over all existentially-quantified atoms that could base-guard x and y) in place of an explicit base-guard. Such an atom $\mathsf{guarded}_{\sigma_{\mathcal{B}}}(x, y)$ is called a *generalized base-guard* since it can express that a pair of elements is guarded without indicating the exact atom that is guarding x and y, and without worrying about other variables that may appear in the guard atom. Note that allowing these generalized base-guards does not increase the expressivity of the logic, but it is convenient to allow them (e.g. in the definition of the k-guardedly linear axioms).

We can now state the following result, which we will prove in the rest of this section, and then extend to show Lemma 4.4:

Proposition A.4. *Every sentence φ in normal form* $\mathsf{BaseGNF}$ *(possibly with generalized base-guards) has base-guarded-interface k-tree-like witnesses where k is the width of φ.*

The result and proof of Proposition A.4 is very similar to Proposition 3.3. However, unlike Proposition 3.3, we do not interpret the distinguished relations in a special way here. This allows us to prove the stronger base-guarded-interface property about the corresponding tree decompositions, which is important in some arguments (e.g., Proposition 4.2 and Theorem 3.13).

We first consider a variant of the GN^k bisimulation game defined earlier in Appendix A.1. The positions in the game are the same as before:

i) partial isomorphisms $f : X \to Y$ or $g : Y \to X$, where $X \subset \mathrm{elems}(\mathcal{F})$ and $Y \subset \mathrm{elems}(\mathcal{G})$ are of size at most k and $\sigma_{\mathcal{B}}$-guarded;

ii) partial rigid homomorphisms $f : X \to Y$ or $g : Y \to X$, where $X \subset \mathrm{elems}(\mathcal{F})$ and $Y \subset \mathrm{elems}(\mathcal{G})$ are of size at most k.

However, the rules of the game are different.

From a type (i) position h, Spoiler must choose a finite subset $X' \subset \mathrm{elems}(\mathcal{F})$ or a finite subset $Y' \subset \mathrm{elems}(\mathcal{G})$, in either case of size at most k, upon which Duplicator must respond by a partial rigid homomorphism h' with domain X' or Y' accordingly that is consistent with h. (This is the same as before).

In a type (ii) position h, Spoiler is only allowed to select some base-guarded subset X' of $\mathsf{Dom}(h)$ (rather than an arbitrary subset of size k), and then the game proceeds from the type (i) position h' obtained by restricting h to this base-guarded subset.

Thus, unlike in the game presented in Appendix A.1, this game strictly alternates between type (ii) positions and base-guarded positions of type (i). We call this a *base-guarded-interface GN^k bisimulation game*, since the interfaces (shared elements) between the domains of consecutive positions must be base-guarded. We can then show the analogue of Proposition A.1 for this variant of the game (note that this time we do not handle the distinguished relations in any special way):

Proposition A.5. *Let $\mathcal{F}$ and $\mathcal{G}$ be sets of facts extending $\mathcal{F}_0$. Let $\varphi(\vec{x})$ be a formula in normal form* BaseGNF *of width at most k.*

If Duplicator has a winning strategy in the base-guarded-interface GN^k bisimulation game between $\mathcal{F}$ and $\mathcal{G}$ starting from a type (i) position $\vec{a} \mapsto \vec{b}$ and $\mathcal{F}$ satisfies $\varphi(\vec{a})$, then $\mathcal{G}$ satisfies $\varphi(\vec{b})$.

Proof. Suppose Duplicator has a winning strategy in the base-guarded-interface GN^k bisimulation game between $\mathcal{F}$ and $\mathcal{G}$. We proceed by induction on the negation depth of φ.

If φ has negation depth 0, then it is a UCQ. We are assuming that $\mathcal{F}, \vec{a}$ satisfies φ, so there is some CQ $\delta = \exists \vec{y}(\chi_1 \wedge \cdots \wedge \chi_j)$ that is satisfied by $\mathcal{F}, \vec{a}$. Hence, there is some $\vec{c} \in \mathrm{elems}(\mathcal{F})$ such that $\mathcal{F}, \vec{a}, \vec{c}$ satisfies $\chi_1 \wedge \cdots \wedge \chi_j$. Because the width of φ is at most k, we know that the combined number of elements in $\vec{a}$ and $\vec{c}$ is at most k. Hence, we can consider the move in the game where Spoiler selects the elements in $\vec{a}$ and $\vec{c}$. Duplicator must respond with some $\vec{d} \in \mathrm{elems}(\mathcal{G})$ such that $\vec{a}\vec{c} \mapsto \vec{b}\vec{d}$ is a partial rigid homomorphism, a type (ii) position. Each conjunct χ_i must be satisfied by $\mathcal{G}, \vec{b}\vec{d}$ since $\vec{a}\vec{c} \mapsto \vec{b}\vec{d}$ is a partial homomorphism with respect to σ. Hence, $\mathcal{G}, \vec{b}\vec{d} \models \delta$, and $\mathcal{G}, \vec{b} \models \varphi$ as desired.

Now assume that φ has negation depth $d > 0$ and is of the special form $\alpha(\vec{x}) \wedge \neg\varphi'(\vec{x}')$. By definition of BaseGNF, it must be the case that $\vec{x}'$ is a sub-tuple of $\vec{x}$, and α is either a base-atom or a generalized base-guard for $\vec{x}'$. Since $\mathcal{F}, \vec{a}$ satisfies φ, we know that $\mathcal{F}, \vec{a}$ satisfies $\alpha(\vec{x})$, which implies (by the inductive hypothesis) that $\mathcal{G}, \vec{b}$ also satisfies $\alpha(\vec{x})$. It remains to show that $\mathcal{G}$ satisfies $\neg\varphi'(\vec{x}')$. Assume for the sake of contradiction that it satisfies $\varphi'(\vec{x}')$. Because $\vec{a} \mapsto \vec{b}$ is a type (i) position, we can consider the move in the game where Spoiler switches the domain to the other set of facts, keeps the same set of elements, and then collapses to the base-guarded elements in the subtuple $\vec{b}'$ of $\vec{b}$ corresponding to $\vec{x}'$ in $\vec{x}$. Let $\vec{a}'$ be the corresponding subtuple of $\vec{a}$. Duplicator must still have a winning strategy from this new type (i) position $\vec{b}' \mapsto \vec{a}'$, so the inductive hypothesis ensures that $\mathcal{F}, \vec{a}'$ satisfies $\varphi'(\vec{x}')$, a contradiction.

Finally consider an arbitrary UCQ-shaped formula φ with negation depth $d > 0$. If $\mathcal{F}, \vec{a} \models \varphi$, then there is some disjunct $\exists \vec{y}(\chi_1 \wedge \cdots \wedge \chi_j)$ in φ and some $\vec{c} \in \mathrm{elems}(\mathcal{F})$ such that $\mathcal{F}, \vec{a}, \vec{c}$ satisfies $\chi_1 \wedge \cdots \wedge \chi_j$. Because the width of φ is at most k, we know that the combined number of elements in $\vec{a}$ and $\vec{c}$ is at most k. Hence, we can consider the move in the game where Spoiler selects the elements in $\vec{a}$ and $\vec{c}$. Duplicator must respond with some $\vec{d} \in \mathrm{elems}(\mathcal{G})$ such that $\vec{a}\vec{c} \mapsto \vec{b}\vec{d}$ is a partial rigid homomorphism, a type (ii) position. Now consider the possible shape of the χ_i. If χ_i is a σ-atom, then it must be satisfied in $\mathcal{G}, \vec{b}\vec{d}$ since $\vec{a}\vec{c} \mapsto \vec{b}\vec{d}$ is a partial homomorphism with respect to σ. Otherwise, χ_i is of the form $\alpha(\vec{x}_i\vec{y}_i) \wedge \neg\varphi'$ where $\vec{x}_i$ and $\vec{y}_i$ are subtuples of variables from $\vec{x}$ and $\vec{y}$ that are actually used by this χ_i, and α is a base-guard for the free variables of φ'. We know that $\mathcal{F}, \vec{a}_i\vec{b}_i \models \alpha(\vec{x}_i\vec{y}_i) \wedge \neg\varphi'$. We can consider Spoiler's restriction of $\vec{a}\vec{c}$ to the subset of elements $\vec{a}_i\vec{b}_i$ that correspond to $\vec{x}_i\vec{y}_i$ and the corresponding restriction of $\vec{b}\vec{d}$ to $\vec{b}_i\vec{d}_i$. This is a valid move to a type (i) position $\vec{a}_i\vec{b}_i \mapsto \vec{b}_i\vec{d}_i$, since $\mathcal{F}, \vec{a}_i\vec{b}_i \models \alpha(\vec{x}_i\vec{y}_i)$ and the definition of BaseGNF requires that α is a base atom. Since Duplicator must still have a winning strategy from this new type (i) position, the previous case implies that this χ_i is also satisfied by $\mathcal{G}, \vec{b}_i\vec{d}_i$. Since this is true for all χ_i in the CQ-shaped formula, $\mathcal{G}, \vec{b}\vec{d} \models \exists \vec{y}(\chi_1 \wedge \cdots \wedge \chi_j)$, and $\mathcal{G}, \vec{b}$ satisfies φ as desired. $\square$

We then use a variant of the unravelling based on this game. The *base-guarded-interface GN^k-unravelling* $\mathcal{F}^k_{\mathcal{B}}$ is defined in a similar fashion to the GN^k-unravelling, except it uses only sequences $\Pi \cap \{X_0 \dots X_n$: for all $i \geq 1$, $X_i \cap X_{i+1}$ is $\sigma_{\mathcal{B}}$-guarded$\}$. This unravelling has an $\mathcal{F}_0$-rooted base-guarded-interface tree decomposition of width $k-1$. Moreover, we can show the analogue of Proposition A.2:

Proposition A.6. *Let $\mathcal{F}$ be a set of facts extending $\mathcal{F}_0$, and let $\mathcal{F}^k_{\mathcal{B}}$ be the base-guarded-interface GN^k-unravelling of $\mathcal{F}$. Then Duplicator has a winning strategy in the base-guarded-interface GN^k bisimulation game between $\mathcal{F}$ and $\mathcal{F}^k_{\mathcal{B}}$.*

Proof. The proof is similar to Proposition A.2. The interesting part of the argument is when Spoiler selects some new elements X' in $\mathcal{F}$ starting from a safe position f (for which there is some π such that $f(a) = [\pi, a]$ for all $a \in \mathsf{Dom}(f)$). We need to show that $[\pi', a']$ for $a' \in X'$ and $\pi' = \pi \cdot X'$ is well-defined in $\mathcal{F}^k_{\mathcal{B}}$. This is well-defined only if the overlap between the elements in π and π' is base-guarded. But because the base-guarded-interface GN^k bisimulation game strictly alternates between type (i) and (ii) positions, Spoiler can only select new elements X' in a type (i) position, so the overlap satisfies this requirement. The remainder of the proof is the same as in Proposition A.2. □

We can conclude the proof of Proposition A.4 as follows. Assume that $\mathcal{F}$ is a set of facts that satisfies φ in normal form of width k. Since $\mathcal{F}$ satisfies φ, Propositions A.5 and A.6 imply that $\mathcal{F}^k_{\mathcal{B}}$ also satisfies $\varphi \in \mathsf{BaseGNF}$. Hence, $\mathcal{F}^k_{\mathcal{B}}$ is a base-guarded-interface k-tree-like witness for φ. This completes the proof of Proposition A.4.

Concluding the proof. We can now use Proposition A.4 to prove Lemma 4.4, which says that there are base-guarded-interface k-tree-like witnesses even when we extend $\Sigma \in \mathsf{BaseCovGNF}$ to Σ' that includes the k-guardedly linear axioms. Recall the formal statement:

> **Lemma 4.4.** The sentence $\Sigma' \wedge \neg Q$ has base-guarded-interface k-tree-like witnesses when taking $k := \max(|\Sigma \wedge \neg Q|, \text{arity}(\sigma \cup \{G\}))$.

Proof. By Proposition 2.4, $\Sigma \wedge \neg Q$ is equivalent to a formula in normal form with width at most $|\Sigma \wedge \neg Q|$. Hence, Proposition A.4 implies that $\Sigma \wedge \neg Q$ has a base-guarded-interface k-tree-like witness for some $k \leq |\Sigma \wedge \neg Q|$. So, in particular, it has one for $k := \max(|\Sigma \wedge \neg Q|, \text{arity}(\sigma \cup \{G\}))$. To prove this lemma, then, it suffices to argue that the k-guardedly linear axioms can also be written in normal form BaseGNF with generalized base-guards and width at most k.

The width of $\mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y)$ is $\text{arity}(\sigma \cup \{G\}) \leq k$. The guardedly total axiom is written in normal form BaseGNF as $\neg \exists xy(\mathsf{guarded}_{\sigma_{\mathcal{B}} \cup \{G\}}(x, y) \wedge \neg(x = y \vee x < y \vee y < x))$ so it has width $\max(2, \text{arity}(\sigma \cup \{G\})) \leq k$. The irreflexive axiom is already written in normal form BaseGNF with width $1 \leq k$. For the k-guardedly transitive axioms, $\psi_l(x, y)$ has width $\max(l + 1, \text{arity}(\sigma \cup \{G\}))$ and $\psi_l(x, x)$ has width $\max(l, \text{arity}(\sigma \cup \{G\}))$, so each of the k-guardedly transitive axioms has width at most k. Overall, this means that the k-guardedly linear restriction can be expressed in normal form BaseGNF with generalized base-guards and width k as required. □

Appendix B. Details about Automata

We now give details of the properties of automata used in the body.

B.1 Closure Properties

We recall some closure properties of 2APT and 1NPT, omitting the standard proofs (Thomas, 1997; Löding, 2011), and following Section 5 of (Benedikt et al., 2016). Note that we state only the size of the automata for each property, but the running time of the procedures constructing these automata is always polynomial in the output size.

First, the automata that we are using are closed under union and intersection (of their languages).

Proposition B.1. 2APT *are closed under union and intersection, with only a polynomial blow-up in the number of states, priorities, and overall size. The same holds of* 1NPT.

For example, this means that if we are given two 2APT $\mathcal{A}_1$ and $\mathcal{A}_2$, then we can construct in PTIME a 2APT $\mathcal{A}$ such that $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

Another important language operation is *projection*. Let L' be a language of trees over a tree signature $\Gamma \cup \{P\}$. The *projection* of L' with respect to P is the language of trees T over Γ such that there is some $T' \in L'$ such that T and T' agree on all unary relations in Γ. Projection is easy for nondeterministic automata since the valuation for P can be guessed by Eve.

Proposition B.2. 1NPT *are closed under projection, with no change in the number of states, priorities, and overall size.*

Finally, complementation is easy for alternating automata by taking the *dual* automaton, obtained by switching conjunctions and disjunctions in the transition function, and incrementing all of the priorities by one.

Proposition B.3. 2APT *are closed under complementation, with no change in the number of states, priorities, and overall size.*

B.2 Proof of Theorem 2.6: Localization

Recall the statement:

> **Theorem 2.6.** Let $\Gamma' := \Gamma \cup \{P_1, \ldots, P_j\}$. Let $\mathcal{A}'$ be a 1NPT on Γ'-trees. We can construct a 2APT $\mathcal{A}$ on Γ-trees such that for all Γ-trees T and nodes v in the domain of T,
>
> $$\mathcal{A}' \text{ accepts } T' \text{ from the root iff } \mathcal{A} \text{ accepts } T \text{ from } v$$
>
> where T' is the Γ'-tree obtained from T by setting $P_1^{T'} = \cdots = P_j^{T'} = \{v\}$. The number of states of $\mathcal{A}$ is linear in the number of states of $\mathcal{A}'$, the number of priorities of $\mathcal{A}$ is linear in the number of priorities of $\mathcal{A}'$, and the overall size of $\mathcal{A}$ is linear in the size of $\mathcal{A}'$. The running time is polynomial in the size of $\mathcal{A}'$, and hence is in PTIME.

This result was known already in the literature. For completeness, we present the construction here, following the presentation from Benedikt et al. (2016).

Let $\mathcal{A}'$ be a 1NPT on Γ'-trees, with state set $Q_{\mathcal{A}'}$, initial state $q^0_{\mathcal{A}'}$, transition function $\delta_{\mathcal{A}'}$, and priority function $\Omega_{\mathcal{A}'}$. Since $\mathcal{A}'$ is nondeterministic, $\delta_{\mathcal{A}'}$ maps each state and label to a disjunction of formulas of the form $(\mathsf{left}, r) \wedge (\mathsf{right}, r')$.

We construct $\mathcal{A}$ as follows. The state set of $\mathcal{A}$ is $\{q_0\} \cup Q_{\mathcal{A}'} \cup (\{\mathsf{left}, \mathsf{right}\} \times Q_{\mathcal{A}'})$. The initial state is q_0. We call states of the form $q \in Q_{\mathcal{A}'}$ *downwards mode states*, and states of the form $(d, q) \in (\{\mathsf{left}, \mathsf{right}\} \times Q_{\mathcal{A}'})$ *upwards mode states*.

We now define the transition function δ of $\mathcal{A}$. In initial state q_0, we set $\delta(q_0, \beta)$ to be:

$$\begin{cases} \bigvee\{(d,r)\wedge(d',r')\wedge(\mathsf{up},(\mathsf{left},q)) : q\in Q_{\mathcal{A}'} \text{ and } (d,r)\wedge(d',r') \text{ is a disjunct in } \delta_{\mathcal{A}'}(q,\beta\cup\{P_1,\ldots,P_j\})\} & \text{if } \mathsf{left}\in\beta \\ \bigvee\{(d,r)\wedge(d',r')\wedge(\mathsf{up},(\mathsf{right},q)) : q\in Q_{\mathcal{A}'} \text{ and } (d,r)\wedge(d',r') \text{ is a disjunct in } \delta_{\mathcal{A}'}(q,\beta\cup\{P_1,\ldots,P_j\})\} & \text{if } \mathsf{right}\in\beta \\ \bigvee\{(d,r)\wedge(d',r') : (d,r)\wedge(d',r') \text{ is in } \delta(q^0_{\mathcal{A}'},\beta\cup\{P_1,\ldots,P_j\})\} & \text{if } \mathsf{left},\mathsf{right}\notin\beta \end{cases}$$

In other words, Eve guesses some $q \in Q_{\mathcal{A}'}$ and some disjunct $(d,r)\wedge(d',r')$ consistent with the transition function of $\delta_{\mathcal{A}'}$ in state q, assuming that the starting label also includes $P_1,\ldots,P_j$. Adam can either (i) challenge her to show that the automaton accepts from state q by moving in direction d or d' and switching to downwards mode state r or r', respectively, or (ii) challenge her to show that she could actually reach state q by switching to upwards mode (d'',q) where d'' is left if the starting node was the left child of its parent and right if it was the right child of its parent. In the special case that the starting node is the root (left and right are not in the label), then q must be $q^0_{\mathcal{A}'}$ and Adam can only challenge downwards.

In upwards mode state (d,r), we set $\delta((d,r),\beta)$ to be:

$$\begin{cases} \bigvee\{(d',r')\wedge(\mathsf{up},(\mathsf{left},q)) : q\in Q_{\mathcal{A}'} \text{ and } (d,r)\wedge(d',r') \text{ or } (d',r')\wedge(d,r) \text{ is a disjunct in } \delta_{\mathcal{A}'}(q,\beta)\} & \text{if } \mathsf{left}\in\beta \\ \bigvee\{(d',r')\wedge(\mathsf{up},(\mathsf{right},q)) : q\in Q_{\mathcal{A}'} \text{ and } (d,r)\wedge(d',r') \text{ or } (d',r')\wedge(d,r) \text{ is a disjunct in } \delta_{\mathcal{A}'}(q,\beta)\} & \text{if } \mathsf{right}\in\beta \\ \bigvee\{(d',r') : (d,r)\wedge(d',r') \text{ or } (d',r')\wedge(d,r) \text{ is in } \delta(q^0_{\mathcal{A}'},\beta)\} & \text{if } \mathsf{left},\mathsf{right}\notin\beta \end{cases}$$

In other words, the upwards mode state (d,r) remembers the state r and child d that the automaton came from. Eve guesses some state q and some disjunct $(d,r)\wedge(d',r')$ or $(d',r')\wedge(d,r)$ in the transition function $\delta_{\mathcal{A}'}$ in state q. That is, she is guessing a possible state in the current node that could have led to her being in state r in the d-child of the current node. Adam can either challenge her on the downwards run from here by moving to the d' child and switching to downwards mode state r', or continue to challenge her upwards by moving up and switching to state (d'',q), where d'' records whether the current node is the left or right child of its parent. In the special case that the current node is the root, then q must be $q^0_{\mathcal{A}'}$ and Adam can only challenge downwards. Note that Adam is not allowed to challenge in direction d, since this is where the automaton came from.

In downwards mode state r, we set $\delta(r,\beta) := \delta_{\mathcal{A}'}(r,\beta)$. In states like this, the automaton is simulating exactly the original automaton $\mathcal{A}'$.

The priority assignment is inherited from $\mathcal{A}'$, namely, we set $\Omega(r) := \Omega_{\mathcal{A}'}(r)$, we set $\Omega((d,r)) := \Omega_{\mathcal{A}'}(r)$, and we set $\Omega(q_0) := 1$.

This concludes the construction of the automaton $\mathcal{A}$ from $\mathcal{A}'$. We must show that when $\mathcal{A}$ is launched from node v in some tree T, it behaves like $\mathcal{A}'$ launched from the root of T', where T' is obtained from T by adding $\{P_1,\ldots,P_j\}$ to the label at v.

A winning strategy ρ' of a nondeterministic automaton like $\mathcal{A}'$ on T' can be viewed as an annotation of T' with states such that (i) the root is annotated with the initial state $q^0_{\mathcal{A}'}$, (ii) if a node v with label β is annotated with q and its left child is annotated with r and right child with r', then $(\mathsf{left},r)\wedge(\mathsf{right},r')$ or $(\mathsf{right},r')\wedge(\mathsf{left},r)$ is a disjunct in $\delta_{\mathcal{A}'}(q,\beta)$, and (iii) the priorities of the states along every branch in T' satisfy the parity condition.

So assume that there is a winning strategy ρ' of $\mathcal{A}'$ on T'. It is not hard to see that this induces a winning strategy ρ for Eve in the acceptance game of $\mathcal{A}$ on T starting from v: Eve guesses in a backwards fashion the part of the run ρ' on the path from v to the root, and then processes the rest of the tree in a normal downwards fashion, using ρ' to drive her choices. Using this strategy, any play is infinite and a suffix of this play (namely, once the play has switched to downward mode) corresponds directly to a suffix of a play in ρ'. Since the priorities in these suffixes are identical, the parity condition must be satisfied, so ρ is winning, and $\mathcal{A}$ accepts T starting from v.

Now suppose Eve has a winning strategy ρ in the acceptance game of $\mathcal{A}$ on T starting from v. Using ρ, we stitch together a winning strategy ρ' of $\mathcal{A}'$ on T'. Recall that such a winning strategy can be viewed as an annotation of T' with states consistent with $\delta_{\mathcal{A}'}$ and satisfying the parity condition on every branch. We construct this annotation starting at v, based on Eve's guess of the state when in q_0. The annotation of the subtree rooted at v is then induced by the plays in ρ that switch immediately to downward mode at v. We can then proceed to annotate the parent u of v, by considering Eve's choice of state when Adam stays in upward mode and moves to the parent u of v. If v is the d-child of u, then the subtree in direction d from u is already annotated; the subtree in the other direction can be annotated by considering the plays when Adam switches to downward mode at u. Continuing in this fashion, we obtain an annotation of the entire tree with states such that $q^0_{\mathcal{A}'}$ is the annotation at the root and the other annotations are consistent with $\delta_{\mathcal{A}'}$ on T' (the annotations are consistent with T' and not T, because in the initial state, Eve's choice of state and disjunct is under the assumption of the extra relations $\{P_1, \dots, P_j\}$ present at v). Every branch in this run tree satisfies the parity condition, since a suffix of the branch corresponds to a suffix of a play in ρ that satisfies the parity condition. Hence, $\mathcal{A}'$ accepts T' from the root.

B.3 Proof of Lemma 3.8: Localized Automata for BaseGNF

Recall the statement:

> **Lemma 3.8.** Let η be in BaseGNF over $\sigma \cup \{Y_1, \dots, Y_s\}$. Let $\mathcal{A}_\eta$ be a localized 2APT for η over $\Sigma^{\text{code}}_{\sigma\cup\{Y_1,\dots,Y_s\},k,l}$-trees.
>
> For $1 \le i \le s$, let $\chi_i := \alpha_i \wedge \neg\psi_i$ be a formula in BaseGNF over σ with the number of free variables in χ_i matching the arity of Y_i, and let $\mathcal{A}_{\chi_i}$ be a localized 2APT for χ_i over $\Sigma^{\text{code}}_{\sigma,k,l}$-trees.
>
> We can construct a localized 2APT $\mathcal{A}_\psi$ for $\psi := \eta[Y_1 := \chi_1, \dots, Y_s := \chi_s]$ over $\Sigma^{\text{code}}_{\sigma,k,l}$-trees in linear time such that the number of states (respectively, priorities) is the sum of the number of states (respectively, priorities) of $\mathcal{A}_\eta, \mathcal{A}_{\chi_1}, \dots, \mathcal{A}_{\chi_s}$.

Let $\psi := \eta[Y_1 := \chi_1, \dots, Y_s := \chi_s]$. To construct $\mathcal{A}_\psi$, take the disjoint union of $\mathcal{A}_\eta$, $\mathcal{A}_{\chi_1}, \dots, \mathcal{A}_{\chi_s}$. Then for each polarity p and localization $\vec{a}/\vec{x}$, set the designated initial state to the initial state for p and $\vec{a}/\vec{x}$ coming from $\mathcal{A}_\eta$. Modify the transition function of $\mathcal{A}_\psi^{p,\vec{a}/\vec{x}}$ so that the automaton starts by simulating $\mathcal{A}_\eta^{p,\vec{a}/\vec{x}}$, but at every node w:

- Eve guesses a valuation for each Y_i at w, which is a (possibly empty) set of facts of the form $Y_{i,\vec{b}}$ where $\vec{b}$ is a subset of names(w).
- Adam can either accept Eve's guesses and continue the simulation of $\mathcal{A}_\eta^{p,\vec{a}/\vec{x}}$, or can challenge one of Eve's assertions of Y_i by launching the appropriate localized version of $\mathcal{A}_{\chi_i}$. That is, if Eve guesses that $Y_{i,\vec{b}}$ holds at w, then Adam could challenge this by launching $\mathcal{A}_{\chi_i}^{+,\vec{b}/\vec{z}}$ starting from w. Likewise, if Eve guesses that $Y_{i,\vec{b}}$ does not hold at w, then Adam could challenge this by launching $\mathcal{A}_{\chi_i}^{-,\vec{b}/\vec{z}}$.

The correctness of this construction relies on the fact that each Y_i is being replaced by a base-guarded formula χ_i, so any Y_i-fact must be about a $\sigma_{\mathcal{B}}$-guarded set of elements. In particular, remember that if this set of elements has cardinality ≤ 1 then the $\sigma_{\mathcal{B}}$-guard may be an equality atom. In any case, these elements must appear together in some node of the tree, so Eve can guess an annotation of the tree that indicates where these Y_i-facts appear.

The proof of correctness follows. We present only the case for $p = +$ and signature $\sigma \cup \{Y\}$, but the case for $p = -$ or multiple Y_i relations is similar.

First, assume that $\mathsf{decode}(T), [v, \vec{a}]$ satisfies ψ, for T a $\Sigma^{\text{code}}_{\sigma,k,l}$-tree. We must show that Eve has a winning strategy ζ in the acceptance game of $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\psi}$ on T starting from v.

For each localization $\vec{b}/\vec{z}$, let $J_{\vec{b}} := \{w \in \mathsf{Dom}(T) : \mathsf{decode}(T), [w, \vec{b}] \models \chi(\vec{z})\}$. For each $w \in J_{\vec{b}}$, let $\zeta_{w,\vec{b}}$ denote Eve's winning strategy in the acceptance game of $\mathcal{A}^{+,\vec{b}/\vec{z}}_{\chi}$ on T starting from w. Likewise, for each $w \notin J_{\vec{b}}$, let $\zeta_{w,\vec{b}}$ denote Eve's winning strategy in the acceptance game of $\mathcal{A}^{-,\vec{b}/\vec{z}}_{\chi}$ on T starting from w. Let T' be T extended with $Y_{\vec{b}}(w)$ for each $\vec{b}$ and w such that $w \in J_{\vec{b}}$. Then $\mathsf{decode}(T'), [v, \vec{a}]$ satisfies η, so Eve has a winning strategy ζ' in the acceptance game of $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\eta}$ on T' starting from v.

We use these strategies to define Eve's strategy ζ in the acceptance game of $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\psi}$ on T starting from v: at each node w, Eve should guess the set $\{Y_{\vec{b}} : w \in J_{\vec{b}}\}$ and use the strategy ζ' (based on the label at w, extended with $\{Y_{\vec{b}} : w \in J_{\vec{b}}\}$). If Adam never challenges her on these guesses, then the play will correspond to a play in ζ', so Eve will win. If Adam challenges her on some guess that $Y_{\vec{b}}$ holds at w (respectively, $Y_{\vec{b}}$ does not hold at w), then the automaton $\mathcal{A}^{+,\vec{b}/\vec{z}}_{\chi}$ (respectively, $\mathcal{A}^{-,\vec{b}/\vec{z}}_{\chi}$) is launched and Eve should switch to the strategy $\zeta_{w,\vec{b}}$ (this is well-defined since Eve is guessing the set based on $J_{\vec{b}}$). But $\zeta_{w,\vec{b}}$ is a winning strategy, so once we switch to this strategy, Eve is guaranteed to win. Hence, ζ is a winning strategy for Eve, as desired.

Now we must prove that if Eve has a winning strategy in the acceptance game of $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\psi}$ on T when launched from v, then $\mathsf{decode}(T), [v, \vec{a}] \models \psi$. We prove the contrapositive. Suppose $\mathsf{decode}(T), [v, \vec{a}] \not\models \psi$. We must give a winning strategy ζ_A for Adam in the acceptance game of $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\psi}$ on T starting from v. Let $J_{\vec{b}} := \{w \in \mathsf{Dom}(T) : \mathsf{decode}(T), [w, \vec{b}] \models \chi(\vec{z})\}$, and let T' be T extended with the valuation for Y such that $Y_{\vec{b}}$ holds at w iff $w \in J_{\vec{b}}$.

Because $\mathsf{decode}(T), [v, \vec{a}] \not\models \psi$, it must be the case that $\mathsf{decode}(T'), [v, \vec{a}] \not\models \eta$. Hence, there is a winning strategy ζ'_A for Adam in the acceptance game of $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\eta}$ on T'. By definition of $J_{\vec{b}}$, for each pair $[w, \vec{b}]$ such that $w \notin J_{\vec{b}}$ (respectively, $w \in J_{\vec{b}}$) it must be the case that $\mathsf{decode}(T), [w, \vec{b}] \not\models \chi$ (respectively, $\mathsf{decode}(T), [w, \vec{b}] \models \chi$), and hence Adam has a winning strategy $\zeta^{w,\vec{b}}_A$ in $\mathcal{A}^{+,\vec{b}/\vec{z}}_{\chi}$ (respectively, $\mathcal{A}^{-,\vec{b}/\vec{z}}_{\chi}$) on T starting from w.

We define ζ_A based on these substrategies. Adam starts by playing according to ζ'_A, and continues using this strategy while the automaton is simulating $\mathcal{A}^{+,\vec{a}/\vec{x}}_{\eta}$ and Eve is only guessing $Y_{\vec{b}}$ at a node w for $w \in J_{\vec{b}}$. If Eve ever deviates from this valuation based on $J_{\vec{b}}$, then Adam challenges this guess and switches to using the appropriate strategy $\zeta^{w,\vec{b}}_A$. In either case, it is clear that the resulting plays will be winning for Adam, so ζ_A is a winning strategy for Adam as desired.

Appendix C. Data Complexity Upper Bounds for Transitivity

We now provide details on the proof of the data complexity upper bounds.

C.1 Details on Quantifier-Rank and Pebble Games

The *quantifier-rank* of a first-order formula φ, written $\mathsf{QR}(\varphi)$ is the number of nested quantifications: that is, a formula with no quantifiers has quantifier-rank 0, while the inductive definition is:

$$\begin{aligned} \mathsf{QR}(\neg\varphi) &= \mathsf{QR}(\varphi) \\ \mathsf{QR}(\varphi_1 \wedge \varphi_2) = \mathsf{QR}(\varphi_1 \vee \varphi_2) &= \max(\mathsf{QR}(\varphi_1), \mathsf{QR}(\varphi_2)) \\ \mathsf{QR}(\exists x\ \varphi) = \mathsf{QR}(\forall x\ \varphi) &= \mathsf{QR}(\varphi) + 1 \end{aligned}$$

We will be interested in showing that two sets of facts I and I' agree on all formulas of quantifier-rank j. This can be demonstrated using the *j-round pebble game on I, I'*. A position in this game is given by a sequence $\vec{p}$ of elements from $\mathrm{elems}(I)$ and a sequence $\vec{p}'$ of the same length as $\vec{p}$ from $\mathrm{elems}(I')$. There are two players, Spoiler and Duplicator, and a round of the game at position $(\vec{p}, \vec{p}')$ proceeds by Spoiler choosing one of the sets of facts (e.g. I) and appending an element from that set of facts to the corresponding sequence (e.g. appending an element from $\mathrm{elems}(I)$ to $\vec{p}$) while Duplicator responds by appending an element from the other set of facts in the other sequence (e.g. appending an element from $\mathrm{elems}(I')$ to $\vec{p}'$). A j-round play of the game is a sequence of j moves as above. Duplicator wins the game if the sequences represent a partial isomorphism: $p_i = p_j$ if and only if $p'_i = p'_j$ and for any relation R, $R(p_{m_1} \dots p_{m_j}) \in I$ if and only if $F(p'_{m_1} \dots p'_{m_j}) \in I'$. A strategy for Duplicator is a response to each move of Spoiler. Such a strategy is *winning* from a given position $\vec{p}, \vec{p}'$ if every j-round play emerging from following the strategy, starting at these positions, is not winning for Spoiler. The following result is well-known (see, e.g. Libkin, 2004):

Proposition C.1. *If there a winning strategy for Duplicator in the j-round pebble game on I, I' starting at $\vec{p}, \vec{p}'$, then for every formula φ of quantifier-rank at most j satisfied by $\vec{p}$ in I, φ is also satisfied by $\vec{p}'$ in I'.*

C.2 Proof of Theorem 3.12: CoNP Data Complexity Bound for QAtc

Recall the statement of Theorem 3.12:

> For any fixed $\mathsf{BaseGNF}$ constraints Σ and UCQ Q, given a finite set of facts $\mathcal{F}_0$, we can decide $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ in CoNP data complexity.

We now prove the theorem. Within this appendix, contrary to the rest of the paper, we will consider logics that feature constants, in which case we will explicitly indicate it.

For a set of facts $\mathcal{F}_0$ over any signature, an *$\mathcal{F}_0, k$-rooted set* consists of $\mathcal{F}_0$ unioned with some sets of facts $T_{\vec{c}}$ for $\vec{c} \in \mathrm{elems}(\mathcal{F}_0)^k$ where the domain of $T_{\vec{c}}$ overlaps with the domain of $\mathcal{F}_0$ only in $\vec{c}$, and for two k-tuples $\vec{c}$ and $\vec{c}'$, the domain of $T_{\vec{c}}$ overlaps with the domain of $T_{\vec{c}'}$ only within $\vec{c} \cap \vec{c}'$.

One can picture such a set as a squid, with $\mathcal{F}_0$ at the root and the $T_{\vec{c}'}$ hanging off as tentacles. Using Proposition 3.3, we can show that if there is a witness to satisfiability of an $\mathsf{BaseGNF}$ sentence in a superset of a set of facts $\mathcal{F}_0$, then there is such an extension that forms an $\mathcal{F}_0, k$-rooted set.

Proposition C.2. *For any set of $\sigma_{\mathcal{B}}$-facts $\mathcal{F}_0$, if a* $\mathsf{BaseGNF}$ *sentence Σ over σ (without constants) is satisfiable by some set of facts containing $\mathcal{F}_0$ with relations R_i^+ interpreted as the transitive closure of R_i, then Σ is satisfied (with the same restriction) in a set of $\sigma_{\mathcal{B}}$-facts which form an $\mathcal{F}_0, k$-rooted set, where k is at most $|\Sigma|$.*

Proof. By Proposition 3.3 there is a set of facts $\mathcal{F}$ consisting of $\sigma_{\mathcal{B}}$-facts whose extension to distinguished facts satisfies Σ, and which has an $\mathcal{F}_0$-rooted tree decomposition. For a child node v of the root of the decomposition of $\mathcal{F}$, its *interface elements* are the values that appear in a fact associated with the root and also in a fact associated with v. For each $\vec{c} \in \mathrm{elems}(\mathcal{F})^k$, let $C_{\vec{c}}$ be the children of the root node of the decomposition whose interface elements are contained in $\vec{c}$. Let $T_{\vec{c}}$ be all $\sigma_{\mathcal{B}}$-facts outside of $\mathcal{F}_0$ that are in a descendant of a node in $C_{\vec{c}}$. It is easy to verify that $\mathcal{F}$ and $T_{\vec{c}} : \vec{c} \in \mathrm{elems}(\mathcal{F}_0)^k$ have the required property. □

Proposition C.2 shows that it suffices to examine witnesses consisting of a rooted set of facts, and a collection of tentacles indexed by k-tuples of the domain of the root. We have control over the size of the root, and also over the index set. But the size of the tree-like tentacles is unbounded. We now show a decomposition result stating that to know what happens in a tree-like set, we will not need to care about the details of the tentacles, but only a small amount of information concerning the sentences that the tentacle satisfies in isolation.

Let $\mathsf{FO}(\sigma)$ denote first-order logic over the signature σ with equality. Let $\mathsf{FO}(\sigma \cup \{d_1 \dots d_k\})$ denote first-order logic over the signature σ with equality and k constants, which will be used to represent the overlap elements. Note that formulas in both $\mathsf{FO}(\sigma)$ and $\mathsf{FO}(\sigma \cup \{d_1 \dots d_k\})$ can use the distinguished relations R_i^+ that are part of σ.

The quantifier-rank of a formula is the maximal number of nested quantifiers; the formal definition is reviewed in Appendix C.1. For any fixed signature ρ, if we fix the quantifier-rank j, we also fix the number of variables that may occur in a formula, and thus there are only finitely many sentences up to logical equivalence. Thus we can let $\mathsf{FO}_j(\rho)$ denote a finite set containing a sentence equivalent to each sentence of quantifier-rank at most j. Given an $\mathcal{F}_0, k$-rooted set $\mathcal{I}$, and number j, the *j-abstraction of $\mathcal{I}$* is the expansion of $\mathcal{F}_0$ with relations $P_\tau(x_1 \dots x_k)$ for each $\tau \in \mathsf{FO}_j(\sigma \cup \{d_1 \dots d_k\})$. We interpret $P_\tau(x_1 \dots x_k)$ by the set of k-tuples $\vec{c}$ such that $T_{\vec{c}}^+$ satisfies τ, where $T_{\vec{c}}^+$ interprets the constants in τ by $\vec{c}$ and the distinguished relations by the appropriate transitive closures. We let $\sigma_{j,k}$ be the signature of the j-abstraction of such structures.

Lemma C.3. For any sentence φ of $\mathsf{FO}(\sigma)$ and any $k \in \mathbb{N}$, there is a $j \in \mathbb{N}$ having the following property:

Let $\mathcal{I}_1$ be an $\mathcal{F}_1, k$-rooted set for some set of $\sigma_{\mathcal{B}}$-facts $\mathcal{F}_1$, and let $\mathcal{I}_1^+$ be its extension with distinguished relations R_i^+ interpreted as the transitive closure of the corresponding base relations R_i. Let $\mathcal{I}_2$ be an $\mathcal{F}_2, k$-rooted set for some set of $\sigma_{\mathcal{B}}$-facts $\mathcal{F}_2$, and $\mathcal{I}_2^+$ the corresponding extension with facts over the distinguished relations. If the j-abstractions of $\mathcal{I}_1$ and $\mathcal{I}_2$ agree on all $\mathsf{FO}(\sigma_{j,k})$ sentences of quantifier-rank at most j, then $\mathcal{I}_1^+$ and $\mathcal{I}_2^+$ agree on φ.

Proof. Let j_φ be the quantifier-rank of φ. We choose $j := j_\varphi \cdot k$. We will show that $\mathcal{I}_1$ and $\mathcal{I}_2$ agree on all formulas of quantifier-rank j_φ. By standard results in finite model theory (reviewed in Appendix C.1), it is sufficient to give a strategy for Duplicator in the j_φ-round standard pebble game for $\mathsf{FO}(\sigma)$ over $\mathcal{I}_1^+$ and $\mathcal{I}_2^+$. With i moves left to play, we will ensure the following invariants on a game position consisting of a sequence $\vec{p}_1 \in \mathcal{I}_1^+$ and $\vec{p}_2 \in \mathcal{I}_2^+$:

- Let $\vec{p_1}'$ be the subsequence of $\vec{p}_1$ that comes from $\mathcal{F}_1$ and let $\vec{p_2}'$ be defined similarly for $\vec{p}_2$ and $\mathcal{F}_2$. Then $\vec{p_1}'$ and $\vec{p_2}'$ should form a winning position for Duplicator in the $(i \cdot k)$-round $\mathsf{FO}(\sigma_{j,k})$ game on the j-abstractions.

- Fix any k-tuple $\vec{c}_1 \in \mathcal{F}_1$ and let $P^1_{\vec{c}_1}$ be the subsequence of $\vec{p}_1$ that lies in $T_{\vec{c}_1}$ within $\mathcal{I}_1^+$. Then if $P^1_{\vec{c}}$ is non-empty, $\vec{c}_1$ also lies in $\vec{p}_1$. Further, letting $\vec{c}_2$ be the corresponding k-tuple to $\vec{c}_1$ in $\vec{p}_2$, and letting $P^2_{\vec{c}_2}$ be the subsequence of $\vec{p}_2$ that lies in $T_{\vec{c}_2}$ within $\mathcal{I}_2^+$, then $P^1_{\vec{c}_1}$ and $P^2_{\vec{c}_2}$ form a winning position in the i-round pebble game on $T^+_{\vec{c}_1}$ and $T^+_{\vec{c}_2}$.

  The analogous property holds for any k-tuple $\vec{c}_2 \in \mathcal{F}_2$.

We now explain the strategy of Duplicator, focusing for simplicity on moves of Spoiler within $\mathcal{I}_1$, with the strategy on $\mathcal{I}_2$ being similar. If Spoiler plays within $\mathcal{F}_1$, Duplicator responds using her strategy for the games on the j-abstractions of $\mathcal{F}_1$ and $\mathcal{F}_2$. It is easy to see that the invariant is preserved.

If Spoiler plays an element within a substructure $T_{\vec{c}_1}$ within $\mathcal{I}_1$ that is already inhabited, then by the inductive invariant, $\vec{c}_1$ is pebbled and there is a corresponding $\vec{c}_2$ in $\mathcal{I}_2$ with substructure $T_{\vec{c}_2}$ of $\mathcal{I}_2$ such that the pebbles within $T_{\vec{c}_2}$ are winning positions in the game on $T^+_{\vec{c}_1}$ and $T^+_{\vec{c}_2}$ with i moves left to play. Thus Duplicator can respond using the strategy in this game from those positions.

Now suppose Spoiler plays an element e_1 within a substructure $T_{\vec{c}_1}$ within $\mathcal{I}_1$ that is not already inhabited. We first use $\vec{c}_1$ as a sequence of plays for Spoiler in the game on the j-abstractions of $\mathcal{I}_1$ and $\mathcal{I}_2$, extending the positions given by $\vec{p_1}'$ and $\vec{p_2}'$. By the inductive invariant, responses of Duplicator exist, and we collect them to get a tuple $\vec{c}_2$. Since a winning strategy in a game preserves atoms, and we have a fact in the j-abstraction corresponding to the j-type of $\vec{c}_1$ in $T^+_{\vec{c}_1}$, we know that $\vec{c}_2$ must satisfy the same j-type in $T^+_{\vec{c}_2}$ that $\vec{c}_1$ does in $T^+_{\vec{c}_1}$. Therefore $\vec{c}_1$ must satisfy the same $\mathsf{FO}(\sigma \cup \{d_1 \dots d_k\})$ sentences of quantifier-rank at most j in $T^+_{\vec{c}_1}$ as $\vec{c}_2$ does in $T^+_{\vec{c}_2}$. Thus Duplicator can use the corresponding strategy to respond to e_1 with an e_2 in $T^+_{\vec{c}_2}$ such that $\{e_1\}$ and $\{e_2\}$ are a winning position in the $(i-1)$-round pebble game on $T^+_{\vec{c}_1}$ and $T^+_{\vec{c}_2}$.

Since the response of Duplicator corresponds to k moves in the game within the j-abstractions, one can verify that the invariant is preserved.

We must verify that this strategy gives a partial isomorphism. Consider a fact F that holds of a tuple $\vec{t}_1$ within $\mathcal{I}_1$, and let $\vec{t}_2$ be the tuple obtained using this strategy in $\mathcal{I}_2$. We first consider the case where F is a $\sigma_{\mathcal{B}}$-fact:

- If $\vec{t}_1$ lies completely within some $T_{\vec{c}_1}$, then the last invariant guarantees that $\vec{t}_2$ lies in some $T_{\vec{c}_2}$. The last invariant also guarantees that $\sigma_{\mathcal{B}}$-facts of $\mathcal{I}_1$ are preserved since such facts must lie in $T_{\vec{c}_1}$, and the corresponding positions are winning in the game between $T^+_{\vec{c}_1}$ and $T^+_{\vec{c}_2}$.

- If $\vec{t}_1$ lies completely within $\mathcal{F}_1$, then the first invariant guarantees that the fact is preserved.

By the definition of a rooted set, the above two cases are exhaustive. We now consider the case where F is of the form $R_i^+(t_1, t_2)$:

- If t_1 and t_2 both lie in some $T_{\vec{c}_1}$, then we reason as in the first case above, since facts over the signature with transitive closures are also preserved in the game between $T^+_{\vec{c}_1}$ and $T^+_{\vec{c}_2}$.

- If t_1 and t_2 are both in $\mathcal{F}_1$, we reason as in the second case above, this time using the fact that transitive closure facts are taken into account in the game on the abstraction.

- If t_1 lies in $T_{\vec{c}_1}$, t_2 lies in $T_{\vec{c}_2}$, then t_1 reaches some $c_i \in \vec{c}_1$, c_i reaches some $c_j \in \vec{c}_2$, and c_j reaches t_2 within $T_{\vec{c}_2}$. Then we use a combination of the first two cases above to conclude that F is preserved. □

From Lemma C.3 we easily obtain:

Corollary C.4. *For any sentence φ and $k \in \mathbb{N}$, there is $j \in \mathbb{N}$ and a sentence φ' in the language $\sigma_{j,k}$ of j-abstractions over σ such that for all sets of $\sigma_{\mathcal{B}}$-facts $\mathcal{F}_0$, an $\mathcal{F}_0, k$-rooted set satisfies φ iff its j-abstraction satisfies φ'.*

Recall that by Proposition C.2, we know it suffices to check for a counterexample to entailment that is an $\mathcal{F}_0, k$-rooted set. Corollary C.4 allows us to do this by guessing an abstraction and checking a first-order property of it. This allows us to finish the proof of Theorem 3.12.

Proof of Theorem 3.12. First, consider the case where the initial set of facts $\mathcal{F}_0$ is restricted to contain only $\sigma_{\mathcal{B}}$-facts. Fixing Q and Σ, we give an NP algorithm for the complement. Let $\varphi = \Sigma \wedge \neg Q$, and $k = |\varphi|$. Let j and φ' be the number and formula guaranteed for φ by Corollary C.4.

Recall that $\mathsf{FO}(\sigma \cup \{d_1 \dots d_k\})$ denotes first-order logic over the signature σ of $\Sigma \wedge \neg Q$ with equality and with k constants, and $\mathsf{FO}_j(\sigma \cup \{d_1 \dots d_k\})$ denotes a finite set containing a sentence equivalent to each sentence of quantifier-rank at most j. Let Types_j be the collection of subsets τ of $\mathsf{FO}_j(\sigma \cup \{d_1 \dots d_k\})$ sentences such that the conjunction of sentences in τ is satisfiable. Note that for any fixed j, the size of $\mathsf{FO}_j(\sigma \cup \{d_1 \dots d_k\})$ is finite, hence the size of Types_j is finite. An element of Types_j can be thought of as a description of a tentacle, telling us everything we need to know for the purposes of the j-abstraction.

Given $\mathcal{F}_0$, guess a function f mapping each k-tuple over $\mathcal{F}_0$ to a $\rho \in \mathsf{Types}_j$. We then check whether for two overlapping k-tuples $\vec{c}$ and $\vec{c}'$, the types $f(\vec{c})$ and $f(\vec{c}')$ are consistent on the atomic formulas that hold on overlapping elements, and whether the atomic formulas of $f(\vec{c})$ contain each fact over $\vec{c}$ in $\mathcal{F}_0$. Finally, for each $\tau \in \mathsf{FO}(\sigma \cup \{d_1 \dots d_k\})$ of quantifier-rank at most j, we form the expansion $\mathcal{F}$ by interpreting P_τ by the set of tuples $\vec{c}$ such that $\tau \in f(\vec{c})$, and we check whether the expansion satisfies φ' with these interpretations, and if so return true.

We argue for correctness. If the algorithm returns true with $\mathcal{F}$ as the witness, then create an $\mathcal{F}_0, k$-rooted set $\mathcal{I}$ by picking for each $\vec{c}$ a structure satisfying the sentences in $f(\vec{c})$ with distinguished elements interpreted by $\vec{c}$. Such a structure exists by satisfiability of $f(\vec{c})$. It assigns atomic formulas consistently on overlapping tuples, by hypothesis, and the atomic formulas it assigns contain each fact of $\mathcal{F}_0$. We let the remaining domain elements be disjoint from the domain of $\mathcal{F}$. Note that by construction, $\mathcal{I}$ has $\mathcal{F}$ as its j-abstraction. By the choice of j and φ', and the observation above, $\mathcal{I}$ satisfies $\Sigma \wedge \neg Q$. Thus $\mathcal{I}$ witnesses that $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ is false.

On the other hand, if $\mathsf{QAtc}(\mathcal{F}_0, \Sigma, Q)$ is false, then by Proposition C.2 we have an $\mathcal{F}_0, k$-rooted set $\mathcal{I}$ that satisfies $\Sigma \wedge \neg Q$. By the choice of j and φ', the j-abstraction of $\mathcal{I}$ satisfies φ'. For each tuple $\vec{c}$ from $\mathcal{F}_0$, the set of formulas of quantifier-rank holding of $\vec{c}$ in the tentacle of $\vec{c}$ must be in Types_j. Hence we can guess f that assigns $\vec{c}$ to this set, and with this f as a witness the algorithm returns true.

When the initial set of facts contains also $\sigma_{\mathcal{D}}$-facts, then we need to ensure that our algorithm guarantees the existence of a witness structure which fulfills these transitivity requirements. We first pre-process the sentence as follows: for each $\sigma_{\mathcal{B}}$ relation R we add a new $\sigma_{\mathcal{B}}$ relation R', and add to our theory Σ the sentences:

$$\forall \vec{x}\, R'(\vec{x}) \to R^+(\vec{x})$$

Note that these sentences are base guarded. Now given an instance $\mathcal{F}_0$ containing facts R^+, we change R^+ to R' and perform the algorithm as before.

□

C.3 Proof of Theorem 3.13: PTIME Data Complexity Bound for QAtr

We now turn to the case where our constraints are restricted to BaseCovFGTGDs and deal with QAtr, not QAtc. Recall that Theorem 3.13 states a PTIME data complexity bound for this case:

> **Theorem 3.13.** For any fixed BaseCovFGTGD constraints Σ and base-covered UCQ Q, given a finite set of facts $\mathcal{F}_0$, we can decide $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ in PTIME data complexity.

The proof will follow from a reduction to traditional QA, similar to the proof of Proposition 4.2:

Proposition C.5. *For any finite set of facts $\mathcal{F}_0$, constraints $\Sigma \in$ BaseCovGNF, and base-covered UCQ Q, we can compute $\mathcal{F}'_0$ and $\Sigma' \in$ GNF in PTIME such that $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ iff $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q)$. Furthermore, if Σ is in BaseCovFGTGD then Σ' is in FGTGD.*

Proof. We define $\mathcal{F}'_0$ and Σ' as follows:

- $\mathcal{F}'_0$ is $\mathcal{F}_0$ together with facts $G(a, b)$ for every pair $a, b \in \mathrm{elems}(\mathcal{F}_0)$ for some fresh binary base relation G, and
- Σ' is Σ together with the k-guardedly-transitive axioms for each distinguished relation, where k is $|\Sigma \wedge \neg Q|$.

These can be constructed in time polynomial in the size of the input.

As discussed in the proof of Lemma 4.4, the k-guardedly transitive axioms (see Section 4) can be written in normal form BaseGNF with width at most k, and hence in GNF.

Now we prove the correctness of the reduction. Suppose $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q)$ holds, so any $\mathcal{F}' \supseteq \mathcal{F}'_0$ satisfying Σ' must satisfy Q. Now consider $\mathcal{F} \supseteq \mathcal{F}_0$ that satisfies Σ and where all R^+ in $\sigma_{\mathcal{D}}$ are transitive. We must show that $\mathcal{F}$ satisfies Q. First, observe that $\mathcal{F}$ satisfies Σ' since the k-guardedly-transitive axioms for R^+ are clearly satisfied for all k when R^+ is transitively closed. Now consider the extension of $\mathcal{F}$ to $\mathcal{F}'$ with additional facts $G(a, b)$ for all $a, b \in \mathrm{elems}(\mathcal{F}_0)$. This must still satisfy Σ': adding these guards means there are additional k-guardedly-transitive requirements on the elements from $\mathcal{F}_0$, but these requirements already hold since R^+ is transitively closed on all elements. Hence, by our initial assumption, $\mathcal{F}'$ must satisfy Q. Since Q does not mention G, the restriction of $\mathcal{F}'$ back to $\mathcal{F}$ still satisfies Q as well. Therefore, $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ holds.

On the other hand, suppose for the sake of contradiction that $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q)$ does not hold, but $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ does. Then there is some $\mathcal{F}' \supseteq \mathcal{F}'_0$ such that $\mathcal{F}'$ satisfies $\Sigma' \wedge \neg Q$, and hence also satisfies $\Sigma \wedge \neg Q$. Since $\Sigma \wedge \neg Q$ is in BaseGNF, Proposition A.4 implies that we can take $\mathcal{F}'$ to be a set of facts that has an $\mathcal{F}'_0$-rooted $(k-1)$-width base-guarded-interface tree decomposition. Let $\mathcal{F}''$ be the result of taking the transitive closure of the distinguished relations in $\mathcal{F}'$. By the Transitivity Lemma (Lemma 4.6), transitively closing like this can only add R^+-facts about pairs of elements that are not base-guarded. Moreover, the Base-Coveredness Lemma (Lemma 4.8) ensures that adding R^+-facts about these non-base-guarded pairs of elements does not affect satisfaction of BaseCovGNF sentences, so $\mathcal{F}''$ must still satisfy $\Sigma \wedge \neg Q$. Restricting $\mathcal{F}''$ to its σ-facts results in an $\mathcal{F}$ where every distinguished relation is transitively closed and where $\Sigma \wedge \neg Q$ is still satisfied, since Σ and Q do not mention relation G. But this contradicts the assumption that $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ holds. This concludes the proof of correctness.

Finally, observe that the k-guardedly-transitive axioms can be written as FGTGDs (in fact, BaseFGTGDs): they are equivalent to the conjunction of FGTGDs of the form

$$\forall x\, y\, x_1 \dots x_{l+1}\, \vec{z}\, \big[\big(x = x_1 \wedge x_{l+1} = y \wedge \\ R^+(x_1, x_2) \wedge \cdots \wedge R^+(x_l, x_{l+1}) \wedge S(x, y, \vec{z})\big) \to R^+(x, y)\big]$$

for all $S \in \sigma_{\mathcal{B}} \cup \{G\}$, $1 \leq l \leq k$, and $R^+ \in \sigma_{\mathcal{D}}$. Therefore, if Σ is in BaseCovFGTGD then Σ' is in FGTGD as claimed. $\square$

Theorem 3.13 easily follows from this:

Proof of Theorem 3.13. Recall that we have fixed constraints Σ in BaseCovFGTGD and a base-covered UCQ Q. We must show PTIME data complexity of $\mathsf{QAtr}(\mathcal{F}_0, \Sigma, Q)$ for any finite initial set of facts $\mathcal{F}_0$. Use Proposition C.5 to construct Σ' from Σ (in constant time, since Σ is fixed) and $\mathcal{F}'_0$ from $\mathcal{F}_0$ (in time polynomial in $|\mathcal{F}_0|$). Since Σ is in BaseCovFGTGD, Σ' is in FGTGD. Therefore, the PTIME data complexity upper bound for QAtr with BaseCovFGTGDs follows from the PTIME data complexity upper bound for QA with FGTGDs (Baget et al., 2011). $\square$

Appendix D. Details about the Chase

The *chase* is a standard database construction (Abiteboul et al., 1995), which applies to a set of facts $\mathcal{F}_0$ and to a set Σ of TGDs, and constructs a set of facts $\mathcal{F}_\infty \supseteq \mathcal{F}_0$, possibly infinite, which satisfies Σ.

To define the chase, we first define the notion of a *trigger* and *active trigger*. A *trigger* for a TGD $\tau : \forall \vec{x}\, \varphi(\vec{x}) \to \exists \vec{y}\, \psi(\vec{x}, \vec{y})$ in a set of facts $\mathcal{F}$ is a homomorphism h from $\varphi(\vec{x})$ to $\mathcal{F}$, i.e., a mapping from $\vec{x}$ to $\mathrm{elems}(\mathcal{F})$ such that the facts of $\varphi(h(\vec{x}))$ are in $\mathcal{F}$. We call h an *active trigger* if h cannot be extended to a homomorphism from $\psi(\vec{x}, \vec{y})$ to $\mathcal{F}$, i.e., there is no mapping h' from $\vec{x} \cup \vec{y}$ to $\mathrm{elems}(\mathcal{F})$ such that $h'(x) = h(x)$ for all $x \in \vec{x}$ and such that the facts of $\psi(h(\vec{x}, \vec{y}))$ are in $\mathcal{F}$.

Given a TGD $\tau : \forall \vec{x}\, \varphi(\vec{x}) \to \exists \vec{y}\, \psi(\vec{x}, \vec{y})$, a set of facts $\mathcal{F}$, and an active trigger h of τ in $\mathcal{F}$, the result of *firing* τ on h is a set of facts $\psi(h(\vec{x}), \vec{b})$ where the $\vec{b}$ are fresh elements called *nulls* that are all distinct and do not occur in $\mathcal{F}$. The application of a *chase round* by a set Σ of TGDs on a set of facts $\mathcal{F}$ is the set of facts $\mathcal{F}'$ obtained by firing simultaneously all TGDs of Σ on all active triggers; formally, it is the union of the set of facts $\mathcal{F}$ and of the facts obtained by firing each TGD τ on each active trigger h of τ in $\mathcal{F}$, using different nulls when firing each TGD.

The *chase* of $\mathcal{F}$ by Σ is the (potentially infinite) set of facts $\mathcal{F}_\infty$ obtained by repeated applications of chase rounds. Formally, we define $\mathcal{F}_i$ for all $i > 0$ as the result of applying a chase round on $\mathcal{F}_{i-1}$, and the chase $\mathcal{F}_\infty$ is the fixpoint of this inflationary operator.

The important points about the chase construction is that the set of facts $\mathcal{F}_\infty$ obtained as a result of the chase is a superset of the initial set of facts $\mathcal{F}_0$, that it satisfies Σ, and that it is created by adding new facts in a way that only overlaps on $\mathrm{elems}(\mathcal{F}_0)$ on elements that occur at an active trigger at a position where they will be exported.

Appendix E. From UCQ to CQ

In this section, we first prove general auxiliary lemmas about reducing from QA problems with UCQs to QA problems with CQs. We first give such a lemma for regular QA, which formalises an existing folklore technique (see, e.g., Section 3.3 of Gottlob and Papadimitriou (2003)). We then adapt this lemma to the various QA notions that we study for distinguished relations. This allows us to revisit the results of Sections 5 and 6 and explain how the proofs in the main text using UCQs can be extended to use only CQs.

The general idea to replace UCQs by CQs is to extend the arity of the relations to include a flag that indicates whether a fact is a "real fact" or a "pseudo-fact": the flag is propagated by the TGDs. (Note that, while the idea is similar, the notions of "real fact" and "pseudo-fact" used in this appendix are not related to those of "genuine fact" and "pseudo-fact" that were used within the proofs of Section 5.) We then add pseudo-facts to the instances to ensure that each UCQ disjunct has a match that involves pseudo-facts. This ensures that we can replace the UCQ by a *conjunction* of the original disjuncts, with an OR on the flag of the match of each disjunct: this OR can be performed using a suitable relation which we add to the instances.

We formalize this general idea in Appendix E.1. We must then tweak the idea to make it work for QA with distinguished relations, as we cannot increase the arity of these relations. When the distinguished relations are linear orders, we first show in Appendix E.2 that we can adapt the idea without increasing the arity of the distinguished relations, provided that the TGDs do not mention these relations and that the query satisfies a condition called *base-domain-coveredness*. When the distinguished relations are transitive or are the transitive closure of another relation, we present in Appendix E.3 a more general technique that allows us, under some conditions on the TGDs and queries, to increase the arity of a subset of the relations (called the *flagged relations*), without changing the arity of the others (which we call the *special relations*, and which include the distinguished relations). We then use these techniques to revisit the results of Section 5 in Appendix E.4 and of Section 6 in Appendix E.5.

E.1 UCQ to CQ for General QA

We first show a translation result from UCQ to CQ for general QA:

Lemma E.1. For any signature σ, TGDs Σ and UCQ Q, one can compute in PTIME a signature σ', TGDs Σ' and CQ Q' such that the QA problem for Σ and Q reduces in PTIME to the QA problem for Σ' and Q' (for combined complexity and for data complexity): namely, given a set of facts $\mathcal{F}_0$ on σ, we can compute in PTIME in $\mathcal{F}_0$, Σ, and Q a set of facts $\mathcal{F}'_0$ on σ' such that $\mathsf{QA}(\mathcal{F}_0, \Sigma, Q)$ holds iff $\mathsf{QA}(\mathcal{F}'_0, \Sigma', Q')$ holds.

Proof. Let σ_{Or} be a constant signature consisting of a ternary relation Or and a unary relation True. We let σ' be the signature obtained from σ by creating one relation R' in σ' for every R in σ with $\mathrm{arity}(R') := \mathrm{arity}(R) + 1$, and further adding the relations of σ_{Or}.

We define Σ' from Σ by considering each TGD $\tau : \forall \vec{x}\ \varphi(\vec{x}) \rightarrow \exists \vec{y}\ \psi(\vec{x}, \vec{y})$, and, letting z be a fresh variable, replacing τ by the TGD $\tau' : \forall \vec{x}\, z\ \varphi'(\vec{x}, z) \rightarrow \exists \vec{y}\ \psi'(\vec{x}, z, \vec{y})$, where φ' and ψ' are obtained from φ and ψ respectively by replacing each σ-atom $R(\vec{w})$ by the σ'-atom $R'(\vec{w}, z)$. As TGD bodies are not empty, the new variable z actually occurs in the new body φ'.

We now describe the construction of Q' from Q. Suppose the UCQ Q is $\bigvee_{1 \leq i \leq m} \exists \vec{x}_i\ Q_i(\vec{x}_i)$, where each Q_i is a conjunction of atoms over σ. Let $z_1, \ldots, z_m$ be fresh variables. For each

$1 \leq i \leq m$, we define a conjunction of atoms $Q'_i(\vec{x}_i, z_i)$ on σ' which is obtained from $Q_i(\vec{x}_i)$ by replacing each σ-atom $R(\vec{w})$ by the σ'-atom $R'(\vec{w}, z_i)$. We now define Q' as:

$$\exists z_1 \ldots z_m \, z'_1 \ldots z'_{m-1} \, \vec{x}_1 \ldots \vec{x}_m$$
$$\mathrm{Or}(z_1, z_2, z'_1) \wedge \mathrm{Or}(z'_1, z_3, z'_2) \wedge \cdots \wedge \mathrm{Or}(z'_{m-2}, z_m, z'_{m-1}) \wedge \mathrm{True}(z'_{m-1}) \wedge \bigwedge_{1 \leq i \leq m} Q'_i(\vec{x}_i, z_i)$$

It is clear that the computation of σ', Σ', and Q' from σ, Σ, and Q is in PTIME.

We now describe the PTIME transformation on input sets of facts. Let $\mathcal{F}_0$ be a set of facts. Letting $\mathfrak{t}$ and $\mathfrak{f}$ be two fresh elements, let $\mathcal{F}_{\mathrm{Or}}$ be the set of facts that contains the fact $\mathrm{True}(\mathfrak{t})$ and the facts $\mathrm{Or}(b, b', b'')$ for all $\{(b, b', b \vee b') \mid b, b' \in \{\mathfrak{f}, \mathfrak{t}\}\}$. Let $\mathcal{F}_{\mathfrak{f}}$ be the set of facts $\{R'(\mathfrak{f}, \ldots, \mathfrak{f}) \mid R' \in \sigma'\}$, and let $(\mathcal{F}_0)_{+\mathfrak{t}}$ be $\{R'(\vec{a}, \mathfrak{t}) \mid R(\vec{a}) \in \mathcal{F}_0\}$. We define $\mathcal{F}'_0 := \mathcal{F}_{\mathrm{Or}} \cup \mathcal{F}_{\mathfrak{f}} \cup (\mathcal{F}_0)_{+\mathfrak{t}}$ which is clearly computable in PTIME.

We now show correctness of the reduction. In the forward direction, consider a counterexample set of facts $\mathcal{F}$ on σ which is a superset of $\mathcal{F}_0$, satisfies Σ, and violates Q: up to renaming we can ensure that $\mathfrak{t}, \mathfrak{f} \notin \mathrm{elems}(\mathcal{F})$. Let us construct a counterexample $\mathcal{F}'$ for $\mathcal{F}'_0$, Σ', and Q', by setting $\mathcal{F}' := \mathcal{F}_{\mathrm{Or}} \cup \mathcal{F}_{+\mathfrak{t}} \cup \mathcal{F}_{\mathfrak{f}}$ where $\mathcal{F}_{\mathrm{Or}}$ and $\mathcal{F}_{\mathfrak{f}}$ are as above and where $\mathcal{F}_{+\mathfrak{t}} := \{R'(\vec{a}, \mathfrak{t}) \mid R(\vec{a}) \in \mathcal{F}\}$.

It is clear that $\mathcal{F}_{+\mathfrak{t}}$, hence $\mathcal{F}'$, is a superset of $\mathcal{F}'_0$, because $\mathcal{F}$ is a superset of $\mathcal{F}_0$. To see why $\mathcal{F}'$ satisfies Σ', consider a match $M' \subseteq \mathcal{F}'$ of the body of a TGD τ' of Σ' in $\mathcal{F}'$. As Σ' does not mention the relations of σ_{Or}, no fact of $\mathcal{F}_{\mathrm{Or}}$ can occur in M'. Now, the facts of $\mathcal{F}_{\mathfrak{f}}$ have $\mathfrak{f}$ as their last element, and those of $\mathcal{F}_{+\mathfrak{t}}$ have $\mathfrak{t}$ as their last element, so, as all atoms of the body of τ' have the same variable at their last element, either $M' \subseteq \mathcal{F}_{\mathfrak{f}}$, or $M' \subseteq \mathcal{F}_{+\mathfrak{t}}$. In the first case, we can find a match of the head of τ in $\mathcal{F}_{\mathfrak{f}}$ (where all variables are mapped to $\mathfrak{f}$), so we conclude that M' is not a violation. In the second case, considering the preimage M of M' in $\mathcal{F}$, it is clear that M is a match of the TGD τ of Σ, so, as $\mathcal{F}$ satisfies Σ, we can extend M to a match of the head of τ in $\mathcal{F}$, yielding a match of the head of τ' in $\mathcal{F}'$, so that again M' cannot be a violation. Hence, $\mathcal{F}'$ satisfies Σ.

Last, to see why $\mathcal{F}'$ violates Q', assume by contradiction that there is a homomorphism from Q' to $\mathcal{F}'$. Notice that, in our construction of $\mathcal{F}'$, the only element $a \in \mathrm{elems}(\mathcal{F}')$ such that $\mathrm{True}(a)$ holds is $a = \mathfrak{t}$. Hence, necessarily, h must map z'_{m-1} to $\mathfrak{t}$. However, as the only Or-facts in $\mathcal{F}'$ are those of $\mathcal{F}_{\mathrm{Or}}$, it is clear that h must map some z_{i_0} to $\mathfrak{t}$. Thus, a suitable restriction of h is a match of $\exists \vec{x}_{i_0} \, Q'_{i_0}(\vec{x}_{i_0}, \mathfrak{t})$ in $\mathcal{F}'$. Now, as all facts in the image of h have $\mathfrak{t}$ as their last element, the image of h must be contained in $\mathcal{F}_{+\mathfrak{t}}$, so we deduce that $\exists \vec{x}_{i_0} \, Q_{i_0}(\vec{x}_{i_0})$ has a match in $\mathcal{F}$, contradicting the fact that $\mathcal{F}$ violates Q. This concludes the forward direction of the correctness proof.

In the backward direction, consider a counterexample set of facts $\mathcal{F}' \supseteq \mathcal{F}'_0$ that satisfies Σ' and violates Q'. Construct the set of facts $\mathcal{F} := \{R(\vec{a}) \mid R'(\vec{a}, \mathfrak{t}) \in \mathcal{F}'\}$. As $(\mathcal{F}_0)_{+\mathfrak{t}} \subseteq \mathcal{F}'_0$, clearly $\mathcal{F}_0 \subseteq \mathcal{F}$. To see why $\mathcal{F}$ satisfies Σ, consider a match $M \subseteq \mathcal{F}$ of the body of some TGD τ of Σ in $\mathcal{F}$, and consider its preimage M' in $\mathcal{F}'$, where all facts have $\mathfrak{t}$ as their last element: M' is a match of the body of $\tau' \in \Sigma'$. Hence, as $\mathcal{F}'$ satisfies Σ', M' extends to a match of the head of τ', and the last elements of all its facts is $\mathfrak{t}$, so we can find a suitable extension in $\mathcal{F}$ as well. Hence, M is not a violation of τ in $\mathcal{F}$, so $\mathcal{F}$ satisfies Σ.

Last, to see why $\mathcal{F}$ violates the UCQ Q, assume by contradiction that the Q has a match in $\mathcal{F}$. This means that there is $1 \leq i_0 \leq m$ such that the disjunct $\exists \vec{x}_{i_0} \, Q_{i_0}(\vec{x}_{i_0})$ has a match M_{i_0} in $\mathcal{F}$. By construction of $\mathcal{F}$, this means that $\exists \vec{x}_{i_0} \, Q'_{i_0}(\vec{x}_{i_0}, \mathfrak{t})$ has a match M'_{i_0} in $\mathcal{F}'$. Now, observe that, for all $1 \leq i \leq m$, there is a match M''_i of $\exists \vec{x}_i \, Q_i(\vec{x}_i, \mathfrak{f})$ in $\mathcal{F}_{\mathfrak{f}}$ obtained by mapping all variables to $\mathfrak{f}$. As $\mathcal{F}_{\mathfrak{f}} \subseteq \mathcal{F}'_0 \subseteq \mathcal{F}'$, the same is true of $\mathcal{F}'$. Now, as $\mathcal{F}_{\mathrm{Or}} \subseteq \mathcal{F}'_0 \subseteq \mathcal{F}'$, we can extend M'_{i_0} and the M''_i for $i \neq i_0$ to a match of Q' in $\mathcal{F}'$ by matching z_{i_0} to $\mathfrak{t}$, every z_i to $\mathfrak{f}$ for $i \neq i_0$, every z'_i for $i' < i_0 - 1$ to $\mathfrak{f}$, and the z'_i for $i' \geq i_0 - 1$ to $\mathfrak{t}$. This contradicts the fact that $\mathcal{F}'$ violates Q'. Hence, $\mathcal{F}$ violates Q and is a counterexample to QA. This concludes the correctness proof. □

E.2 UCQ to CQ for QAlin

We first adapt the general QA result in Lemma E.1 to QAlin, which avoids increasing the arity of the distinguished relations that are interpreted as linear orders. In order to avoid increasing the arity of the distinguished relations, we will ban distinguished relations in the dependencies, and require *base-domain-coveredness* of the query (which is a weakening of base-coveredness):

Definition E.2. *A CQ Q is* base-domain-covered *if every variable x occurring in a distinguished atom in Q also occurs in a base atom in Q.*

We can now state the following variant of Lemma E.1:

Lemma E.3. For any signature σ (partitioned in base relations $\sigma_{\mathcal{B}}$ and distinguished relations $\sigma_{\mathcal{D}}$), TGDs Σ on $\sigma_{\mathcal{B}}$, and base-domain-covered UCQ Q, one can compute in PTIME a signature σ' partitioned as $\sigma'_{\mathcal{B}} \cup \sigma_{\mathcal{D}}$, TGDs Σ' on $\sigma'_{\mathcal{B}}$, and a base-domain-covered CQ Q' such that the QAlin problem for Σ and Q reduces in PTIME to the same problem for Σ' and Q', in the sense of Lemma E.1.

Further, if Σ is BaseIDs then Σ' also is; if Σ is empty then Σ' also is; if Q is base-covered then Q' also is.

Proof. We first preprocess the input UCQ Q without loss of generality to remove any disjuncts where some distinguished relation $<_i$ is not a partial order (i.e., it has a cycle): the rewritten Q is equivalent to the original one for QAlin because the removed disjuncts can never be entailed because of the semantics of distinguished relations.

We now adapt the proof of Lemma E.1. The definition of σ_{Or} is unchanged, and we define $\sigma'_{\mathcal{B}} := \sigma''_{\mathcal{B}} \cup \sigma_{\mathrm{Or}}$ where $\sigma''_{\mathcal{B}}$ is defined by increasing the arity of the relations from $\sigma_{\mathcal{B}}$ as before. The definition of Σ' is unchanged. It easy to see that if Σ is empty then so is Σ', and if Σ consists of BaseIDs (i.e., there are no repetitions of variables in the body and in the head, and only one body fact) then this is still the case of Σ'.

The definition of Q' is unchanged except that we do not rewrite distinguished relations: as the query is base-domain-covered, each fresh variable z_i that we add must actually occur in Q'_i, and the base-domain-coveredness (resp. base-coveredness) of Q' is easy to see from that of Q.

We modify the definition of $\mathcal{F}_{\mathrm{Or}}$ to complete each distinguished relation to a total order on $\mathcal{F}_{\mathrm{Or}}$ (e.g., create $\mathfrak{f} <_i \mathfrak{t}$ for all distinguished relations $<_i$). We define $\mathcal{F}_{\mathfrak{f}}$ in a new fashion. First, letting m be the maximal number of variables of a disjunct of Q, for each distinguished relation $<_i$, we create fresh values $\mathfrak{f}^i_1, \ldots, \mathfrak{f}^i_m$, and create facts $\mathfrak{f}^i_1 <_i \cdots <_i \mathfrak{f}^i_m$ in $\mathcal{F}_f$. Second, we create all facts $R'(\vec{f}, \mathfrak{f})$ where $R' \in \sigma_{\mathcal{B}}$ and $\vec{f}$ is any tuple of the $\mathfrak{f}^i_j$. Having defined $\mathcal{F}_{\mathfrak{f}}$, we then define $(\mathcal{F}_0)_{+\mathfrak{t}}$ as the union of $\{R'(\vec{a}, \mathfrak{t}) \mid R(\vec{a}) \in \mathcal{F}_0 \land R \in \sigma_{\mathcal{B}}\}$ and of the $\sigma_{\mathcal{D}}$-facts of $\mathcal{F}_0$ kept as-is; then we define $\mathcal{F}'_0 := \mathcal{F}_{\mathrm{Or}} \cup (\mathcal{F}_0)_{+\mathfrak{t}} \cup \mathcal{F}_{\mathfrak{f}}$.

To adapt the forward direction of the correctness proof, let us consider a counterexample $\mathcal{F}$ to $\mathsf{QAlin}(\mathcal{F}_0, \Sigma, Q)$ with suitably interpreted distinguished relations. We define $\mathcal{F}'' := \mathcal{F}'_0 \cup \mathcal{F}_{+\mathfrak{t}} \cup \mathcal{F}_{\mathfrak{f}}$, with $\mathcal{F}_{\mathfrak{f}}$ as above and $\mathcal{F}_{+\mathfrak{t}}$ defined as above by adding $\mathfrak{t}$ to base facts and keeping distinguished facts as-is. It is clear that each distinguished relation in $\mathcal{F}''$ is a partial order, because this is true in isolation in $\mathcal{F}_{\mathfrak{f}}$, in $\mathcal{F}_{\mathrm{Or}}$, and in $\mathcal{F}_{+\mathfrak{t}}$; further $\mathcal{F}_{\mathfrak{f}}$ and $\mathcal{F}_{\mathrm{Or}}$ overlap only on $\mathfrak{f}$, $\mathcal{F}_{\mathrm{Or}}$ and $\mathcal{F}_{+\mathfrak{t}}$ only overlap on $\mathfrak{t}$, and $\mathcal{F}_{\mathfrak{f}}$ and $\mathcal{F}_{+\mathfrak{t}}$ do not overlap at all. Thus, we can define $\mathcal{F}'$ from $\mathcal{F}''$ by completing each distinguished relation to be a total order.

We now explain how the correctness argument of the forward direction is adapted. Clearly $\mathcal{F}' \supseteq \mathcal{F}'_0$. To see why $\mathcal{F}'$ satisfies Σ', as Σ' does not involve the distinguished relations, we reason as in Lemma E.1 to deduce that a match is either included in $\mathcal{F}_{\mathfrak{f}}$ or in $\mathcal{F}_{+\mathfrak{t}}$: the first case is similar as a head match can be found in $\mathcal{F}_{\mathfrak{f}}$ by definition, and the second case is unchanged. To see why $\mathcal{F}'$ violates Q', we show as in Lemma E.1 that there is a match h of some Q'_{i_0} to $\mathcal{F}'$ that maps all *base* facts of Q'_{i_0} to facts with $\mathfrak{t}$ as their last element. Now, the only distinguished facts where each individual element occurs in $\mathcal{F}'$ in base facts of this form are the ones from $\mathcal{F}_{+\mathfrak{t}}$, that we constructed from $\mathcal{F}$, which violated Q_{i_0}; hence, we can conclude using the fact that Q is base-domain-covered.

We now explain how the backward direction of the correctness proof is adapted. We construct $\mathcal{F}$ as the disjoint union of $\{R(\vec{a}) \mid R'(\vec{a}, \mathfrak{t}) \in \mathcal{F} \wedge R' \in \sigma''_{\mathcal{B}}\}$ and of the distinguished facts of $\mathcal{F}'$ kept as-is. It is clear that $\mathcal{F}'$ suitably interprets the distinguished relations, because its restriction to $\sigma_{\mathcal{D}}$ is the same as $\mathcal{F}$, which does. Again we have $\mathcal{F} \supseteq \mathcal{F}_0$. The fact that $\mathcal{F}$ satisfies Σ is as before, except that the arity of distinguished facts is not changed in M', and the witness head facts of M' in $\mathcal{F}'$ may include distinguished facts, in which case they are found as-is in $\mathcal{F}$.

To see that $\mathcal{F}$ violates Q, we reuse the argument of Lemma E.1. The only new point that is needed is that the new $\mathcal{F}_{\mathfrak{f}}$ can still be used to find matches of any Q'_{i_0} with the last variable mapped to $\mathfrak{f}$, but this is easy to see from that construction (also recall our initial preprocessing of Q to eliminate disjuncts where some distinguished relation was not a partial order). This concludes the proof. □

E.3 UCQ to CQ for QA with Special Relations

We now adapt Lemma E.1 so it can be applied to both QAtr and QAtc. We partition the signature into two sets of relations: *flagged relations*, whose arity will be increased as in the proof of Lemma E.1 above, and *special relations*, whose arity we will not increase. This partition is different from that of the rest of the paper, where we had base and distinguished relations. In this section, the special relations must include all distinguished relations (so that we do not increase their arity), but they may also include base relations. In particular, for QAtc, the base relations of which we are taking the transitive closure must themselves be special (indeed, we cannot increase their arities).

We prove a generalization of Lemma E.1 to the setting with flagged relations and special relations, and where we also allow logical constraints on special relations that go beyond the TGDs allowed in Lemma E.1: in particular this will allow us to impose transitivity and transitive closure requirements. The tradeoff is that we will need to impose a restriction on the TGDs and CQ: intuitively, when we use the special relations, we must also use the flagged relations (so we cannot simply make all relations special).

We first define the constraints that we will allow on the special relations:

Definition E.4. *On a signature $\sigma := \sigma_{\mathcal{F}} \cup \sigma_{\mathcal{S}}$ partitioned into flagged and special relations, a* special constraint set Θ *is a set of logical constraints on $\sigma_{\mathcal{S}}$ involving any of the following:*

- *Disjunctive inclusion dependencies on $\sigma_{\mathcal{S}}$;*
- Transitivity assertions*, i.e., assertions that some binary relation in $\sigma_{\mathcal{S}}$ is transitive (i.e., a special kind of TGD);*

- Transitive closure assertions*, i.e., assertions that some binary relation in $\sigma_{\mathcal{S}}$ is the transitive closure of another binary relation in $\sigma_{\mathcal{S}}$.*

Hence, special constraint sets can be used to express the semantics of distinguished relations in the QAtr and QAtc problems : remember that distinguished relations are always special, and for QAtc the base relations of which we are taking the transitive closure are also special.

In addition to the special constraint set Θ, our result will allow us to write TGDs Σ, and the negation of a CQ, like in Lemma E.1. However, in exchange for the freedom of keeping special relations binary, we need to impose a condition on the TGDs and on the CQ, which we call *flagged-reachability*. Intuitively, the goal of this condition is to ensure that we can discriminate between matches of special relations in the query or in dependency bodies that use facts annotated by $\mathfrak{t}$, versus the matches whose facts are annotated by $\mathfrak{f}$. Indeed, this information cannot be seen on the special relations because they do not carry the flag.

Definition E.5. *Let G be the graph over the atoms of Q where two atoms are connected iff they share a variable. A CQ Q is* flagged-reachable *if any special atom $A(x, y)$ in Q has a path to some flagged atom $B(\vec{z})$ in G. A UCQ is flagged-reachable if all of its disjuncts are. A TGD is* flagged-reachable *if its body is.*

The flagged-reachable restriction suffices to ensure that matches of special relations in queries and rule bodies must correspond to $\mathfrak{f}$ or to non-$\mathfrak{f}$ elements, by looking at the flagged facts to which the special relations must be connected. We can thus show:

Lemma E.6. For any signature $\sigma := \sigma_{\mathcal{F}} \cup \sigma_{\mathcal{S}}$, flagged-reachable TGDs Σ, special constraint set Θ on $\sigma_{\mathcal{S}}$, and flagged-reachable CQ Q, one can compute in PTIME a signature $\sigma' := \sigma'_{\mathcal{F}} \cup \sigma_{\mathcal{S}}$, TGDs Σ' and CQ Q' such that the QA problem for $\Sigma \cup \Theta$ and Q reduces in PTIME to the QA problem for $\Sigma' \cup \Theta$ and Q' (for combined complexity and for data complexity): namely, given a set of facts $\mathcal{F}_0$ on σ, we can compute in PTIME in $\mathcal{F}_0$, Σ, Θ and Q a set of facts $\mathcal{F}'_0$ on σ' such that $\mathsf{QA}(\mathcal{F}_0, \Sigma \cup \Theta, Q)$ holds iff $\mathsf{QA}(\mathcal{F}'_0, \Sigma' \cup \Theta, Q')$ holds.

Further, the following properties transfer from Σ to Σ': being IDs; being BaseIDs; being empty. Further, if some relation in σ is not mentioned in Q then Q' does not mention it either.

Proof. We amend the proof of Lemma E.1. We first explain the change in the construction. The definition of σ_{Or} is unchanged, and we define $\sigma'_{\mathcal{F}} := \sigma''_{\mathcal{F}} \cup \sigma_{\mathrm{Or}}$ where $\sigma''_{\mathcal{F}}$ is defined by increasing the arity of the relations from $\sigma_{\mathcal{F}}$ (like σ' from σ in Lemma E.1). The definition of Σ' is unchanged except that the special relations are not rewritten; note that flagged-reachability and non-emptiness of the bodies ensure that they always contain a flagged relation, so that the variable that we add indeed occurs in the body (but it may not occur in the head). Clearly Σ' is still flagged-reachable. Further, it is clear that, if Σ consists of IDs or BaseIDs, then Σ' also does, as there is still only one fact in the body (and in the case of BaseID it is still a flagged fact), and there are no repetitions of variables. It is further clear that if Σ is empty then Σ' also is.

The definition of Q' is unchanged except that special relations are not rewritten; again, flagged-reachability of the query ensures that each fresh variable z_i indeed occurs in Q'_i, and the condition on Q' clearly follows from the condition on Q. We define $\mathcal{F}_{\mathfrak{f}} := \{R(\mathfrak{f}, \ldots, \mathfrak{f}) \mid R \in \sigma'_{\mathcal{F}} \cup \sigma_{\mathcal{S}}\}$, and we define $(\mathcal{F}_0)_{+\mathfrak{t}}$ as the union of $\{R'(\vec{a}, \mathfrak{t}) \mid R(\vec{a}) \in \mathcal{F}_0 \wedge R \in \sigma_{\mathcal{F}}\}$ and of the $\sigma_{\mathcal{S}}$-facts of $\mathcal{F}_0$ kept as-is; then we define $\mathcal{F}'_0 := \mathcal{F}_{\mathrm{Or}} \cup (\mathcal{F}_0)_{+\mathfrak{t}} \cup \mathcal{F}_{\mathfrak{f}}$ as before.

We now explain how to modify the correctness proof. For the forward direction, consider a counterexample $\mathcal{F}$ to $\mathsf{QA}(\mathcal{F}_0, \Sigma \cup \Theta, Q)$, assuming without loss of generality that $\mathfrak{t}, \mathfrak{f} \notin \mathcal{F}$. We define $\mathcal{F}' := \mathcal{F}'_0 \cup \mathcal{F}_{+\mathfrak{t}} \cup \mathcal{F}_{\mathfrak{f}}$, with $\mathcal{F}_{\mathfrak{f}}$ as above and $\mathcal{F}_{+\mathfrak{t}}$ defined as above by adding $\mathfrak{t}$ to flagged facts and keeping special facts as-is.

To verify that this construction for the forward direction is correct, we must first show that $\mathcal{F}'$ satisfies the constraints Θ on $\sigma_{\mathcal{S}}$. For disjunctive inclusion dependencies τ, letting M be a match of the body, as Θ does not mention the facts of $\mathcal{F}_{\mathrm{Or}}$, we must have $M \in \mathcal{F}_{\mathfrak{f}}$ or $M \in \mathcal{F}_{+\mathfrak{t}}$. Now, in the first case, by construction $\mathcal{F}_{\mathfrak{f}}$ must contain a match of some head disjunct of τ, and in the second case, by considering M in $\mathcal{F}$ which satisfies τ, we can also extend the match in $\mathcal{F}_{+\mathfrak{t}}$ hence in $\mathcal{F}'$. For transitivity assertions, we reason in the same way: seeing them as a TGD with a connected body that does not mention the relations of σ_{Or}, we deduce again that any match of them must be within $\mathcal{F}_{\mathfrak{f}}$ or within $\mathcal{F}_{+\mathfrak{t}}$, which allows us to conclude. The same reasoning works for transitive closure assertions. Hence, $\mathcal{F}'$ satisfies Θ.

Now, we must verify the other conditions on $\mathcal{F}'$, for which we adapt the proof of Lemma E.1. In particular, observe that $\mathcal{F}'$ is still a superset of $\mathcal{F}'_0$. To check that there are no violations of Σ', consider a match M' of a TGD τ' of Σ'; as before M' includes no fact of $\mathcal{F}_{\mathrm{Or}}$, and the *flagged facts* $M'_{\mathcal{B}}$ of M' are either included in $\mathcal{F}_{+\mathfrak{t}}$ or in $\mathcal{F}_{\mathfrak{f}}$ depending on their last element. Now, as τ' is flagged-reachable, we observe that the *special facts* $M'_{\mathcal{D}}$ of M' must be connected to the flagged facts, so that, as $\mathcal{F}_{+\mathfrak{t}}$ and $\mathcal{F}_{\mathfrak{f}}$ have disjoint domains and no facts connect them except σ_{Or}-facts which do not occur in τ', it must again be the case that the *entire match* M' is either included in $\mathcal{F}_{\mathfrak{f}}$ or in $\mathcal{F}_{+\mathfrak{t}}$. So we can conclude as before (in the second case, the preimage M of M' in $\mathcal{F}$ is defined without changing the special facts, but the same reasoning applies).

To check that $\mathcal{F}'$ violates Q', as before we reason by contradiction and deduce that $\exists \vec{x}_{i_0} Q'_{i_0}(\vec{x}_{i_0}, \mathfrak{t})$ holds in $\mathcal{F}'$. Now, its match M' in $\mathcal{F}'$ must consist of a match $M'_{\mathcal{B}}$ of flagged facts of M' with $\mathfrak{t}$ as their last position (so they are in $\mathcal{F}_{+\mathfrak{t}}$) and a match $M'_{\mathcal{D}}$ of special facts. As in the previous paragraph, we now use the fact that Q, hence Q'_{i_0}, is flagged-reachable, so we must also have $\mathrm{elems}(M'_{\mathcal{D}}) \subseteq \mathrm{elems}(M'_{\mathcal{B}})$. Hence, we have $M' \subseteq \mathcal{F}''$, and we deduce as before (except that we do not increase the arity of special facts) that the preimage M of M' is a match of Q in $\mathcal{F}$ and conclude by contradiction. This proves the correctness of the forward direction.

For the backward direction, we build $\mathcal{F}$ as the disjoint union of $\{R(\vec{a}) \mid R'(\vec{a}, \mathfrak{t}) \in \mathcal{F} \wedge R' \in \sigma'_{\mathcal{F}}\}$ and of the special facts of $\mathcal{F}'$ kept as-is. It is clear that $\mathcal{F}'$ satisfies Θ, because its restriction to $\sigma_{\mathcal{S}}$ is the same as $\mathcal{F}$, which satisfies Θ. Again we have $\mathcal{F} \supseteq \mathcal{F}_0$. The fact that $\mathcal{F}$ satisfies Σ is as before, except that the arity of special facts is not changed in M', and the witness head facts of M' in $\mathcal{F}'$ may include special facts, in which case they are found as-is in $\mathcal{F}$. The fact that $\mathcal{F}$ violates Q is exactly as before, which concludes the correctness proof. □

E.4 Revisiting the Results of Section 5

We now apply the results of Appendix E.2 and E.3 to show the results of Section 5 with a CQ instead of a UCQ.

E.4.1 THEOREM 5.1

We apply Lemma E.6 by picking as special relations E and E^+, and taking all others to be flagged relations. The special constraint set Θ asserts that E^+ is the transitive closure of E. We observe that the $\mathsf{BaseIDs}$ Σ' created in the proof are flagged-reachable, because their bodies always consist of base facts. Now, we observe that the UCQ Q' is also flagged-reachable: the E-atoms in Q-generated disjuncts are connected to the corresponding atom $R'(\vec{x}, e, f)$, the E-facts in E-path length restriction disjuncts are connected to an R'-fact, and the E-facts in DID satisfaction disjuncts are connected to the $\mathsf{Witness}_\tau$-fact.

Hence, we can deduce from Lemma E.6 that Theorem 5.1 extends to QAtc with CQs, both for data complexity and combined complexity.

E.4.2 THEOREM 5.4

It is immediate to observe that the BaseIDs Σ' do not mention the distinguished relations $<$, and we have already observed in the proof that the UCQ that we define is base-covered, hence base-domain-covered, so we deduce from Lemma E.3 that Theorem 5.4 also extends to QAlin with CQs, for data and combined complexity.

E.4.3 PROPOSITION 5.3

We let E and E^+ be the special relations, let all others be flagged relations, and let the special constraint set Θ assert that E^+ is the transitive closure of E. It is easy to observe that the query defined in the proof is flagged-reachable (in particular thanks to the C_χ-atom in the E-path length restriction disjunct). As the constraints are empty, their image by Lemma E.6 also is, so we deduce that the data complexity lower bound of Proposition 5.3 still applies to QAtc with CQs.

E.4.4 PROPOSITION 5.6

As the UCQ defined in the proof is base-covered, hence base-domain-covered, and the constraints are empty, we deduce from Lemma E.3 that the lower bound still applies to QAlin with CQs.

E.5 Revisiting the Results of Section 6

We again apply the results of Appendix E.2 and E.3 to show the results of Section 6 with a CQ instead of a UCQ.

E.5.1 THEOREM 6.2

We use Lemma E.6. The special relations are S^+, K_i, and K'_i, and the other relations are flagged. We let the special constraint set Θ consist of the three DIDs used in the proof, and of the assertion that S^+ is transitive. We let Σ consist of the two other dependencies used in the proof, which are inclusion dependencies, and are flagged-reachable. Applying Lemma E.6, we can reduce our undecidable QA problem to QA on signature with S^+ as its only distinguished relation, with a CQ which does not mention the distinguished relation S^+ (because the original CQ did not), and with constraints comprising Θ (so a transitivity assertion for S^+, as well as DIDs), the translation of Σ (which are inclusion dependencies, so also DIDs). This allows us to conclude that the QA problem studied in Theorem 6.2 is indeed undecidable.

E.5.2 THEOREM 6.1

As before, we use Lemma E.6, and pick S' as the only flagged relation, and let all other relations be special relations. We let Σ consist of the one ID applying only to S'; it is flagged-reachable. The disjuncts of the CQ that we write are always flagged-reachable, as they are connected and always include a S'-fact. The special constraint set Θ consists of all other IDs used in the proof, and of the assertion that S^+ is the transitive closure of S. Applying Lemma E.6, we reduce the QA problem with a UCQ to QA for Θ (so IDs plus the transitive closure assertion on the one distinguished relation S^+), for the translation of Σ' (which are IDs), and a CQ which still does not use the one distinguished relation S^+ of the new signature. This establishes the result of Theorem 6.1.

E.5.3 THEOREM 6.3

We use Lemma E.3. We check that, indeed, the constraints do not mention the distinguished relation, and that the two UCQ disjuncts which mention these relations are base-domain-covered. Hence, the translation shows undecidability of QAlin for BaseIDs and a CQ, concluding the proof of Theorem 6.3.

References

Abiteboul, S., & Duschka, O. M. (1998). Complexity of answering queries using materialized views. In *Proc. of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*.

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of Databases*. Addison-Wesley.

Afrati, F., Li, C., & Pavlaki, V. (2008). Data exchange in the presence of arithmetic comparisons. In *Proc. of 11th International Conference on Extending Database Technology (EDBT)*.

Amarilli, A., Benedikt, M., Bourhis, P., & Vanden Boom, M. (2016). Query answering with transitive and linear-ordered data. In *Proc. of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*.

Andréka, H., Németi, I., & van Benthem, J. (1998). Modal languages and bounded fragments of predicate logic. *J. Philosophical Logic*, *27*(3).

Artale, A., Ryzhikov, V., & Kontchakov, R. (2012). Dl-lite with attributes and datatypes. In *Proc. of 20th European Conference on Artificial Intelligenc (ECAI)*.

Baget, J.-F., Leclère, M., Mugnier, M.-L., & Salvat, E. (2009). Extending decidable cases for rules with existential variables. In *Proc. of the 21st International Joint Conference on Artificial Intelligence IJCAI*.

Baget, J.-F., Mugnier, M.-L., Rudolph, S., & Thomazo, M. (2011). Walking the complexity lines for generalized guarded existential rules. In *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*.

Baget, J., Bienvenu, M., Mugnier, M., & Rocher, S. (2015). Combining existential rules and transitivity: Next steps. In *Proc. of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI*.

Bárány, V., & Bojańczyk, M. (2012). Finite satisfiability for guarded fixpoint logic. *IPL*, *112*(10), 371–375.

Bárány, V., ten Cate, B., & Otto, M. (2012). Queries with guarded negation. *Proc. VLDB Endow.*, *5*(11), 1328–1339.

Bárány, V., ten Cate, B., & Segoufin, L. (2011). Guarded negation. In *Proc. of 38th International Colloquium on Automata Languages and Programming (ICALP)*.

Benedikt, M., Bourhis, P., & Vanden Boom, M. (2016). A step up in expressiveness of decidable fixpoint logics. In *Proc. of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS)*.

Börger, E., Grädel, E., & Gurevich, Y. (1997). *The Classical Decision Problem*. Perspectives in Mathematical Logic. Springer.

Bourhis, P., Krötzsch, M., & Rudolph, S. (2015). Reasonable highly expressive query languages. In *Proc. of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*.

Bourhis, P., Morak, M., & Pieris, A. (2013). The impact of disjunction on query answering under guarded-based existential rules. In *Proc. of the 23rd International Joint Conference on Artificial Intelligence, (IJCAI)*.

Calvanese, D., De Giacomo, G., & Vardi, M. Y. (2005). Decidable containment of recursive queries. *Theor. Comput. Sci.*, *336*(1), 33–56.

Calvanese, D., Eiter, T., & Ortiz, M. (2009). Regular path queries in expressive description logics with nominals. In *Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*.

Calvanese, D., Lembo, D., Lenzerini, M., & Rosati, R. (2006). Data complexity of query answering in description logics. In *Proc. of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*.

Ganzinger, H., Meyer, C., & Veanes, M. (1999). The two-variable guarded fragment with transitive relations. In *Proc. of the 14th Annual IEEE Symposium on Logic in Computer Science (LICS)*.

Glimm, B., Lutz, C., Horrocks, I., & Sattler, U. (2008). Conjunctive query answering for the description logic SHIQ. *J. Artif. Intell. Res.*, *31*, 157–204.

Gottlob, G., & Papadimitriou, C. (2003). On the complexity of single-rule Datalog queries. *Inf. Comp.*, *183*.

Gottlob, G., Pieris, A., & Tendera, L. (2013). Querying the guarded fragment with transitivity. In *Proc. of the 40th International Colloquium on Automata, Languages, and Programming (ICALP)*.

Grädel, E. (2002). Guarded fixed point logics and the monadic theory of countable trees. *Theor. Comput. Sci.*, *288*(1), 129–152.

Grädel, E., & Walukiewicz, I. (1999). Guarded fixed point logic. In *Proc. of the 14th Annual IEEE Symposium on Logic in Computer Science (LICS)*.

Gutiérrez-Basulto, V., Ibáñez-García, Y. A., Kontchakov, R., & Kostylev, E. V. (2015). Queries with negation and inequalities over lightweight ontologies. *J. Web Sem.*, *35*, 184–202.

Horrocks, I., & Sattler, U. (1999). A description logic with transitive and inverse roles and role hierarchies. *Journal of Logic and Computation*, *9*(3), 385–410.

Kieronski, E. (2011). Decidability issues for two-variable logics with several linear orders. In *Proc. of the 20th Annual Conference of the EACSL (CSL)*.

Kieroński, E., & Tendera, L. (2007). On finite satisfiability of the guarded fragment with equivalence or transitive guards. In *Proc. of the International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*.

Krötzsch, M., & Rudolph, S. (2007). Conjunctive queries for $\mathcal{EL}$ with composition of roles. In *Proc. of the 2007 International Workshop on Description Logics (DL2007)*.

Libkin, L. (2004). *Elements of Finite Model Theory*. Springer.

Löding, C. (2011). Automata on infinite trees. `http://www.automata.rwth-aachen.de/~loeding/inf-tree-automata.pdf`.

Ortiz, M., Rudolph, S., & Šimkus, M. (2011). Query answering in the Horn fragments of the description logics SHOIQ and SROIQ. In *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*.

Ortiz, M., Rudolph, S., & Šimkus, M. (2010). Query answering is undecidable in DLs with regular expressions, inverses, nominals and counting. Tech. rep., Technische Universität Wien.

Ortiz, M., & Simkus, M. (2012). Reasoning and query answering in description logics. In *Proc. of the 8th International Summer School on emantic Technologies for Advanced Query Answering (Reasoning Web)*.

Ortiz de la Fuente, M. (2010). *Query Answering in Expressive Description Logics*. Ph.D. thesis, Technische Universität Wien.

Otto, M. (2001). Two variable first-order logic over ordered domains. *J. Symb. Log.*, *66*(2), 685–702.

Rosati, R. (2007). The limits of querying ontologies. In *Proc. of the 11th International Conference on Database Theory (ICDT)*.

Savkovic, O., & Calvanese, D. (2012). Introducing datatypes in dl-lite. In *Proc. of the 20th European Conference on Artificial Intelligence (ECAI)*.

Stefanoni, G., Motik, B., Krötzsch, M., & Rudolph, S. (2014). The complexity of answering conjunctive and navigational queries over OWL 2 EL knowledge bases. *J. Artif. Intell. Res.*, *51*, 645–705.

Szpilrajn, E. (1930). Sur l'extension de l'ordre partiel. *Fundamenta Mathematicae*, *16*(1), 386–389.

Szwast, W., & Tendera, L. (2004). The guarded fragment with transitive guards. *Annals of Pure and Applied Logic*, *128*(13), 227 – 276.

Thomas, W. (1997). Languages, Automata, and Logic. In Rozenberg, G., & Salomaa, A. (Eds.), *Handbook of Formal Languages*. Springer-Verlag.

Vardi, M. Y. (1998). Reasoning about the past with two-way automata. In *Proc. of the 25th International Colloquium on Automata, Languages and Programming (ICALP)*.