Finite-Cliquewidth Sets of Existential Rules

Toward a General Criterion for Decidable yet Highly Expressive Querying

Thomas Feller
Technische Universität Dresden, Germany

Tim S. Lyon
Technische Universität Dresden, Germany

Piotr Ostropolski-Nalewaja
Technische Universität Dresden, Germany
University of Wrocław, Poland

Sebastian Rudolph
Technische Universität Dresden, Germany

Abstract

In our pursuit of generic criteria for decidable ontology-based querying, we introduce *finite-cliquewidth sets* (**fcs**) of existential rules, a model-theoretically defined class of rule sets, inspired by the *cliquewidth* measure from graph theory. By a generic argument, we show that **fcs** ensures decidability of entailment for a sizable class of queries (dubbed *DaMSOQs*) subsuming conjunctive queries (CQs). The **fcs** class properly generalizes the class of finite-expansion sets (**fes**), and for signatures of arity $\leq$2, the class of bounded-treewidth sets (**bts**). For higher arities, **bts** is only indirectly subsumed by **fcs** by means of reification. Despite the generality of **fcs**, we provide a rule set with decidable CQ entailment (by virtue of first-order-rewritability) that falls outside **fcs**, thus demonstrating the incomparability of **fcs** and the class of finite-unification sets (**fus**). In spite of this, we show that if we restrict ourselves to single-headed rule sets over signatures of arity $\leq$2, then **fcs** subsumes **fus**.

2012 ACM Subject Classification Theory of computation → Description logics; Theory of computation → Database query languages (principles); Theory of computation → Logic and databases; Mathematics of computing → Graph theory

Keywords and phrases existential rules, TGDs, cliquewidth, treewidth, bounded-treewidth sets, finite-unification sets, first-order rewritability, monadic second-order logic, datalog

Digital Object Identifier 10.4230/LIPIcs.ICDT.2023.18

Related Version *Full Version*: `https://arxiv.org/abs/2209.02464` [15]

Funding Work supported by the European Research Council (ERC) Consolidator Grant 771779 *A Grand Unified Theory of Decidability in Logic-Based Knowledge Representation* (DeciGUT).

1 Introduction

The problem of querying under *existential rules*[1] (henceforth often shortened to *rules*) is a popular topic in the research fields of database theory and knowledge representation. For arbitrary rule sets, query entailment is undecidable [9], motivating research into expressive fragments for which decidability can be regained.

[1] Existential rules are also referred to as *tuple-generating dependencies* (TGDs) [1], *conceptual graph rules* [31], Datalog$^{\pm}$ [20], and $\forall\exists$-*rules* [3] in the literature.

A theoretical tool which has not only proven useful for describing querying methods, but also for identifying classes of rule sets with decidable query entailment, is the *chase* [4]. Given a database $\mathcal{D}$ and a rule set $\mathcal{R}$, the (potentially non-terminating) chase procedure yields a so-called *universal model* [14]. Universal models satisfy exactly those queries entailed by $\mathcal{D}$ and $\mathcal{R}$ and thus allow for the reduction of query entailment to query evaluation.

Rule sets admitting finite universal models (a property equivalent to being a *finite-expansion set* [3] or *core-chase terminating* [14]) are particularly well-behaved. But even in cases where universal models are necessarily infinite, they may still be sufficiently "tame" to allow for decidable query entailment, as is the case with the class of *bounded-treewidth sets* (**bts**) [3]. A rule set $\mathcal{R}$ qualifies as **bts** *iff* for every database $\mathcal{D}$, there exists a universal model of $\mathcal{D}$ and $\mathcal{R}$ whose *treewidth* (a structural measure originating from graph theory) is bounded by some $n \in \mathbb{N}$. The **bts** class subsumes class of finite expansion sets (**fes**), and gives rise to decidable query entailment for a sizable family of concrete, syntactically defined classes of rule sets deploying more or less refined versions of guardedness [3, 7, 23].

While **bts** is a fairly general class, it still fails to contain rather simple rule sets; e.g., the rule set $\mathcal{R}^{\infty}_{\text{tran}}$, consisting of the two rules

$$\mathtt{E}(x,y) \to \exists z \mathtt{E}(y,z) \quad \text{and} \quad \mathtt{E}(x,y) \land \mathtt{E}(y,z) \to \mathtt{E}(x,z),$$

falls outside the **bts** class. To give an idea as to why this holds, consider any database $\mathcal{D}$ containing an E-fact (e.g., $\mathtt{E}(\mathtt{a},\mathtt{b})$): the chase will yield a universal model resembling the transitive closure of an infinite E-path, a clique-like structure of infinite treewidth. What is more, not only does $\mathcal{R}^{\infty}_{\text{tran}}$ fail to be **bts**, it also fails to fall under the other generic decidability criteria: it is neither **fus** (described below) nor *finitely controllable* (which would guarantee the existence of a finite "countermodel" for each non-entailed query). In spite of this, results for description logics confirm the decidability of (conjunctive) query entailment over $\mathcal{R}^{\infty}_{\text{tran}}$ (see [18, 24]), incentivizing a generalization of the decidability criteria mentioned above.

On a separate, but related note: the **bts** class is incomparable with the class of *finite-unification sets* (**fus**) [3], another class giving rise to decidable query entailment thanks to *first-order rewritability.* Such rule sets may feature non-guarded rules [8]; for instance, they may include *concept products* [6, 30], which create a biclique linking instances of two unary predicates, as in

$$\mathtt{Elephant}(x) \land \mathtt{Mouse}(y) \to \mathtt{BiggerThan}(x,y).$$

The above examples demonstrate a crucial weakness of the **bts** class, namely, its inability to tolerate universal models exhibiting "harmless" unbounded clique-like structures. Opportunely, the graph-theoretic notion of *cliquewidth* [13] overcomes this problem, while retaining most of the desirable properties associated with the notion of treewidth. Inspired by this less mainstream, but more powerful concept, we set out to introduce *finite-cliquewidth sets* (**fcs**) of rules. As the original cliquewidth notion is tailored to finite undirected graphs, some (not entirely straightforward) generalizations are necessary to adapt it to countable instances, while at the same time, preserving its advantageous properties.

Our contributions can be summarized as follows:

- We introduce an abstract framework showing how to utilize specific types of model-theoretic measures to establish the decidability of query entailment for a comprehensive class of queries (dubbed *datalog / monadic second-order queries*, or *DaMSOQs* for short) that significantly extends the class of conjunctive queries.

- Generalizing the eponymous notion from finite graph theory, we introduce the model-theoretic measure of *cliquewidth* for countable instances over arbitrary signatures. Based on our framework, we show that the derived notion of *finite-cliquewidth sets* guarantees decidability of DaMSOQ entailment. In particular, we demonstrate that $\mathcal{R}^{\infty}_{\mathrm{tran}}$ is indeed **fcs**, thus showing that **fcs** incorporates rule sets outside **bts** and **fus**.
- We compare the **fcs** and **bts** classes, obtaining: (good news) for binary signatures, **fcs** subsumes **bts**, which (bad news) does not hold for higher-arity signatures, but (relieving news) **fcs** still "indirectly subsumes" higher-arity **bts** through reification.
- We compare the **fcs** and **fus** classes, obtaining: (good news) for sets of single-headed rules over signatures of arity ≤ 2, **fcs** subsumes **fus**, which (bad news) does not hold for multi-headed rules, but (relieving news) this could not be any different as there are **fus** rule sets for which DaMSOQ entailment is undecidable.

For space reasons, we defer technical details and most proofs to the extended version [15].

2 Preliminaries

Syntax and formulae. We let $\mathfrak{T}$ denote a set of *terms*, defined as the union of three countably infinite, mutually disjoint sets of *constants* $\mathfrak{C}$, *nulls* $\mathfrak{N}$, and *variables* $\mathfrak{V}$. We use $\mathtt{a}$, $\mathtt{b}$, $\mathtt{c}$, ... (occasionally annotated) to denote constants, and use x, y, z, ... (occasionally annotated) to denote both nulls and variables. A *signature* Σ is a finite set of *predicate symbols* (called *predicates*), which are capitalized ($\mathtt{E}$, $\mathtt{R}$, ...). Throughout the paper, we assume a fixed signature Σ, unless stated otherwise. For each predicate $\mathtt{R} \in \Sigma$, we denote its *arity* with $\mathsf{ar}(\mathtt{R})$. We say Σ is *binary* if it contains only predicates of arity ≤ 2. We assume Σ to contain a special, "universal" unary predicate $\top$, assumed to hold of every term.[2] An *atom* over Σ is an expression of the form $\mathtt{R}(\vec{t})$, where $\vec{t}$ is an $\mathsf{ar}(\mathtt{R})$-tuple of terms. If $\vec{t}$ consists only of constants, then $\mathtt{R}(\vec{t})$ is a *ground atom*. An *instance* $\mathcal{I}$ over Σ is a (possibly countably infinite) set of atoms over constants and nulls, whereas a *database* $\mathcal{D}$ is a finite set of ground atoms. The *active domain* $\mathsf{dom}(\mathcal{X})$ of a set of atoms $\mathcal{X}$ is the set of terms appearing in the atoms of $\mathcal{X}$. Moreover, as instances over binary signatures can be viewed as directed edge-labelled graphs, we often use graph-theoretic terminology when discussing such objects.

Homomorphisms. Given sets $\mathcal{X}$, $\mathcal{Y}$ of atoms, a *homomorphism* from $\mathcal{X}$ to $\mathcal{Y}$ is a mapping $h : \mathsf{dom}(\mathcal{X}) \to \mathsf{dom}(\mathcal{Y})$ that satisfies: (i) $\mathtt{R}(h(\vec{t})) \in \mathcal{Y}$, for all $\mathtt{R}(\vec{t}) \in \mathcal{X}$, and (ii) $h(\mathtt{a}) = \mathtt{a}$, for each $\mathtt{a} \in \mathfrak{C}$. $\mathcal{X}$ and $\mathcal{Y}$ are *homomorphically equivalent* (written $\mathcal{X} \equiv \mathcal{Y}$) *iff* homomorphisms exist from $\mathcal{X}$ to $\mathcal{Y}$ and from $\mathcal{Y}$ to $\mathcal{X}$. A homomorphism h is an *isomorphism iff* it is bijective and h^{-1} is also a homomorphism. An instance $\mathcal{I}'$ is an *induced sub-instance* of an instance $\mathcal{I}$ *iff* (i) $\mathcal{I}' \subseteq \mathcal{I}$ and (ii) if $\mathtt{R}(\vec{t}) \in \mathcal{I}$ and $\vec{t} \subseteq \mathsf{dom}(\mathcal{I}')$, then $\mathtt{R}(\vec{t}) \in \mathcal{I}'$.

Existential rules. An *(existential) rule* ρ is a first-order sentence $\forall \vec{x}\vec{y}\, \phi(\vec{x}, \vec{y}) \to \exists \vec{z}\, \psi(\vec{y}, \vec{z})$, where $\vec{x}$, $\vec{y}$, and $\vec{z}$ are mutually disjoint tuples of variables, and both the *body* $\phi(\vec{x}, \vec{y})$ and the *head* $\psi(\vec{y}, \vec{z})$ of ρ (denoted with $\mathsf{body}(\rho)$ and $\mathsf{head}(\rho)$, respectively) are conjunctions (possibly empty, sometimes seen as sets) of atoms over the indicated variables. The *frontier* $fr(\rho)$ of ρ is the set of variables $\vec{y}$ shared between the body and the head. We often omit the universal quantifiers prefixing existential rules. A rule ρ is (i) *n-ary iff* all predicates appearing in ρ

[2] Assuming the presence of such a built-in *domain predicate* $\top$ does not affect our results, but it allows for a simpler and more concise presentation.

are of arity at most n, (ii) *single-headed iff* $\mathsf{head}(\rho)$ contains a single atom, (iii) *datalog iff* ρ does not contain an existential quantifier (otherwise *non-datalog*). We call a finite set of existential rules $\mathcal{R}$ a *rule set.* Satisfaction of a rule ρ (a rule set $\mathcal{R}$) by an instance $\mathcal{I}$ is defined as usual and is written $\mathcal{I} \models \rho$ ($\mathcal{I} \models \mathcal{R}$, respectively). Given a database $\mathcal{D}$ and a rule set $\mathcal{R}$, we define the pair $(\mathcal{D}, \mathcal{R})$ to be a *knowledge base*, and define an instance $\mathcal{I}$ to be a *model* of $(\mathcal{D}, \mathcal{R})$, written $\mathcal{I} \models (\mathcal{D}, \mathcal{R})$, *iff* $\mathcal{D} \subseteq \mathcal{I}$ and $\mathcal{I} \models \mathcal{R}$. A model $\mathcal{I}$ of $(\mathcal{D}, \mathcal{R})$ is called *universal iff* there is a homomorphism from $\mathcal{I}$ into every model of $(\mathcal{D}, \mathcal{R})$.

Rule application and Skolem chase. A rule $\rho = \phi(\vec{x}, \vec{y}) \to \exists \vec{z} \psi(\vec{y}, \vec{z})$ is *applicable* to an instance $\mathcal{I}$ *iff* there is a homomorphism h from $\phi(\vec{x}, \vec{y})$ to $\mathcal{I}$. We then call (ρ, h) a *trigger* of $\mathcal{I}$. The *application* of a trigger (ρ, h) in $\mathcal{I}$ yields the instance $\mathbf{Ch}(\mathcal{I}, \rho, h) = \mathcal{I} \cup \bar{h}(\psi(\vec{y}, \vec{z}))$, where $\bar{h}$ extends h, mapping each variable z from $\vec{z}$ to a null denoted $z_{\rho, h(\vec{y})}$. Note that this entails $\mathbf{Ch}(\mathbf{Ch}(\mathcal{I}, \rho, h), \rho, h') = \mathbf{Ch}(\mathcal{I}, \rho, h)$ whenever $h(\vec{y}) = h'(\vec{y})$. Moreover, applications of different rules or the same rule with different frontier-mappings are independent, so their order is irrelevant. Hence we can define the parallel one-step application of all applicable rules as

$$\mathbf{Ch}_1(\mathcal{I}, \mathcal{R}) = \bigcup_{\rho \in \mathcal{R},\, (\rho, h) \text{ trigger of } \mathcal{I}} \mathbf{Ch}(\mathcal{I}, \rho, h).$$

Then, we define the *(breadth-first) Skolem chase sequence* by letting $\mathbf{Ch}_0(\mathcal{I}, \mathcal{R}) = \mathcal{I}$ and $\mathbf{Ch}_{i+1}(\mathcal{I}, \mathcal{R}) = \mathbf{Ch}_1(\mathbf{Ch}_i(\mathcal{I}, \mathcal{R}), \mathcal{R})$, ultimately obtaining the *Skolem chase* $\mathbf{Ch}_\infty(\mathcal{I}, \mathcal{R}) = \bigcup_{i \in \mathbb{N}} \mathbf{Ch}_i(\mathcal{I}, \mathcal{R})$. We note that the Skolem chase of a countable instance is countable, as is the number of overall rule applications performed to obtain $\mathbf{Ch}_\infty(\mathcal{I}, \mathcal{R})$.

(Unions of) conjunctive queries and their entailment. A *conjunctive query* (CQ) is a formula $q(\vec{y}) = \exists \vec{x}\, \phi(\vec{x}, \vec{y})$ with $\phi(\vec{x}, \vec{y})$ a conjunction (sometimes written as a set) of atoms over the variables from $\vec{x}, \vec{y}$ and constants. The variables from $\vec{y}$ are called *free.* A *Boolean* CQ (or BCQ) is a CQ with no free variables. A *union of conjunctive queries* (UCQ) $\psi(\vec{y})$ is a disjunction of CQs with free variables $\vec{y}$. We will treat UCQs as sets of CQs. A BCQ $q = \exists \vec{x}\, \phi(\vec{x})$ is satisfied in an instance $\mathcal{I}$ if there exists a homomorphism from $\phi(\vec{x})$ to $\mathcal{I}$. $\mathcal{I}$ satisfies a union of BCQs if it satisfies at least one of its disjuncts. An instance $\mathcal{I}$ and rule set $\mathcal{R}$ *entail* a BCQ $q = \exists \vec{x}\, \phi(\vec{x})$, written $(\mathcal{I}, \mathcal{R}) \models q$ *iff* $\phi(\vec{x})$ maps homomorphically into every model of $\mathcal{I}$ and $\mathcal{R}$. This coincides with the existence of a homomorphism from $\phi(\vec{x})$ into any universal model of $\mathcal{I}$ and $\mathcal{R}$ (e.g., the Skolem chase $\mathbf{Ch}_\infty(\mathcal{I}, \mathcal{R})$).

Rewritings and finite-unification sets. Given a rule set $\mathcal{R}$ and a CQ $q(\vec{y})$, we say that a UCQ $\psi(\vec{y})$ is a *rewriting* of $q(\vec{y})$ under the rule set $\mathcal{R}$ *iff* for any database $\mathcal{D}$ and any tuple of its elements $\vec{\mathsf{a}}$ the following holds: $\mathbf{Ch}_\infty(\mathcal{D}, \mathcal{R}) \models q(\vec{\mathsf{a}})$ *iff* $\mathcal{D} \models \psi(\vec{\mathsf{a}})$. A rule set $\mathcal{R}$ is a *finite-unification set* (**fus**) *iff* for every CQ, there exists a UCQ rewriting [3]. This property is also referred to as *first-order rewritability.* If a rule set $\mathcal{R}$ is **fus**, then for any given CQ $q(\vec{y})$, we fix one of its rewritings under $\mathcal{R}$ and denote it with $\mathsf{rew}_{\mathcal{R}}(q(\vec{y}))$.

Treewidth and bounded-treewidth sets. Let $\mathcal{I}$ be an instance. A *tree decomposition* of $\mathcal{I}$ is a (potentially infinite) tree $T = (V, E)$, where:

- $V \subseteq 2^{\mathsf{dom}(\mathcal{I})}$, that is, each node $X \in V$ is a set of terms of $\mathcal{I}$, and $\bigcup_{X \in V} X = \mathsf{dom}(\mathcal{I})$,
- for each $\mathtt{R}(t_1, \ldots, t_n) \in \mathcal{I}$, there is an $X \in V$ with $\{t_1, \ldots, t_n\} \subseteq X$,
- for each term t in $\mathcal{I}$, the subgraph of T induced by the nodes X with $t \in X$ is connected.

The *width* of a tree decomposition is set to be the maximum over the sizes of all its nodes minus 1, if such a maximum exists; otherwise, it is set to ∞. Last, we define the *treewidth* of an instance $\mathcal{I}$ to be the minimal width among all of its tree decompositions, and denote the treewidth of $\mathcal{I}$ as $\mathsf{tw}(\mathcal{I})$. A set of rules $\mathcal{R}$ is a *bounded-treewidth set* (**bts**) *iff* for any database $\mathcal{D}$, there is a universal model for $(\mathcal{D}, \mathcal{R})$ of finite treewidth.[3]

3 A Generic Decidability Argument

In this section, we provide an abstract framework for establishing decidability of entailment for a wide range of queries, based on certain model-theoretic criteria being met by the considered rule set. We first recall the classical notion of a (Boolean) datalog query, and after, we specify the class of queries considered in our framework.

▶ **Definition 1.** *Given a signature* Σ*, a* (Boolean) datalog query q *over* Σ *is represented by a finite set* $\mathcal{R}_\mathsf{q}$ *of datalog rules with predicates from* $\Sigma_{\mathrm{EDB}} \uplus \Sigma_{\mathrm{IDB}}$ *where* $\Sigma_{\mathrm{EDB}} \subseteq \Sigma$ *and* $\Sigma_{\mathrm{IDB}} \cap \Sigma = \emptyset$ *such that (i)* Σ_{EDB}*-atoms do not occur in rule heads of* $\mathcal{R}_\mathsf{q}$*, and (ii)* Σ_{IDB} *contains a distinguished nullary predicate* `Goal`*. Given an instance* $\mathcal{I}$ *and a datalog query* q*, we say that* q holds in $\mathcal{I}$ *(*$\mathcal{I}$ satisfies q*), written* $\mathcal{I} \models \mathsf{q}$*, iff* $\mathtt{Goal} \in \mathbf{Ch}_\infty(\mathcal{I}, \mathcal{R}_\mathsf{q})$*. Query entailment is defined via satisfaction as usual.* ◀

Datalog queries can be equivalently expressed as sentences in second-order logic (with the Σ_{IDB} predicates quantified over) or in least fixed-point logic (LFP). For our purposes, the given formulation is the most convenient; e.g., it makes clear that the second-order entailment problem $(\mathcal{D}, \mathcal{R}) \models \mathsf{q}$ reduces to the first-order entailment problem $(\mathcal{D}, \mathcal{R} \cup \mathcal{R}_\mathsf{q}) \models \mathtt{Goal}$.

▶ **Definition 2.** *A* datalog/MSO query (DaMSOQ) *over a signature* Σ *is a pair* (q, Ξ)*, where* q *is a datalog query over* Σ *and* Ξ *is a monadic second-order (MSO)*[4] *sentence equivalent to* q*. Satisfaction and entailment of DaMSOQs is defined via any of their constituents.* ◀

Consequently, DaMSOQs are the (semantic) intersection of datalog and MSO queries. While representing DaMSOQs as a pair (q, Ξ) is logically redundant, it is purposeful and necessary: the below decision procedure requires both constituents as input and it is not generally possible to compute one from the other. Arguably, the most comprehensive, known DaMSOQ fragment that is well-investigated and has a syntactic definition is that of *nested monadically defined queries*, a very expressive yet computationally manageable formalism [29], subsuming (unions of) Boolean CQs, *monadic datalog queries* [10], *conjunctive 2-way regular path queries* [16], and nested versions thereof (e.g. *regular queries* [27] and others [5]).

▶ **Definition 3.** *Let* Σ *be a finite signature. A* width measure *(for* Σ*) is a function* w *mapping every countable instance over* Σ *to a value from* $\mathbb{N} \cup \{\infty\}$*. We call* w MSO-friendly *iff there exists an algorithm that, taking a number* $n \in \mathbb{N}$ *and an MSO sentence* Ξ *as input,*

- *never terminates if* Ξ *is unsatisfiable, and*
- *always terminates if* Ξ *has a model* $\mathcal{I}$ *with* $\mathsf{w}(\mathcal{I}) \leq n$*.* ◀

[3] The term "*finite*-treewidth set" would be more fitting and in line with our terminology, but we stick to the established name. Also, the **bts** notion is not used entirely consistently in the literature; it sometimes refers to structural properties of a specific type of chase. The "semantic **bts**" notion adopted here subsumes all the others.

[4] For an introduction to monadic second-order logic, see [13, Section 1.3].

As an unsophisticated example, note that the *expansion* function $\mathsf{expansion} : \mathcal{I} \mapsto |\mathsf{dom}(\mathcal{I})|$, mapping every countable instance to the size of its domain, is an MSO-friendly width measure: there are up to isomorphism only finitely many instances with n elements, which can be computed and checked. As a less trivial example, the notion of treewidth has also been reported to fall into this category [3].

▶ **Definition 4.** *Let* w *be a width measure. A rule set* $\mathcal{R}$ *is called a* finite-w set *iff for every database* $\mathcal{D}$*, there exists a universal model* $\mathcal{I}^*$ *of* $(\mathcal{D}, \mathcal{R})$ *satisfying* $\mathsf{w}(\mathcal{I}^*) \in \mathbb{N}$. ⌟

Note that the finite width required by this definition does not need to be uniformly bounded: it may depend on the database and thus grow beyond any finite bound. This is already the case when using the $\mathsf{expansion}$ measure from above, giving rise to the class of *finite-expansion sets* (**fes**), coinciding with the notion of *core-chase terminating* rule sets [14].

▶ **Theorem 5.** *Let* w *be an MSO-friendly width measure and let* $\mathcal{R}$ *be a* finite-w set*. Then, the entailment problem* $(\mathcal{D}, \mathcal{R}) \models (\mathfrak{q}, \Xi)$ *for any database* $\mathcal{D}$ *and DaMSOQ* $(\mathfrak{q}, \Xi)$ *is decidable.*

Proof. We prove decidability by providing two semi-decision procedures: one terminating whenever $(\mathcal{D}, \mathcal{R}) \models (\mathfrak{q}, \Xi)$, the other terminating whenever $(\mathcal{D}, \mathcal{R}) \not\models (\mathfrak{q}, \Xi)$. Then, these two procedures, run in parallel, constitute a decision procedure.

- Detecting $(\mathcal{D}, \mathcal{R}) \models (\mathfrak{q}, \Xi)$. We note that $(\mathcal{D}, \mathcal{R}) \models (\mathfrak{q}, \Xi)$ *iff* $(\mathcal{D}, \mathcal{R}) \models \mathfrak{q}$ *iff* $(\mathcal{D}, \mathcal{R} \cup \mathcal{R}_\mathfrak{q}) \models \mathtt{Goal}$. The latter is a first-order entailment problem. Thanks to the completeness of first-order logic [19], we can recursively enumerate all the consequences of $(\mathcal{D}, \mathcal{R} \cup \mathcal{R}_\mathfrak{q})$ and terminate as soon as we find $\mathtt{Goal}$ among those, witnessing entailment of the query.
- Detecting $(\mathcal{D}, \mathcal{R}) \not\models (\mathfrak{q}, \Xi)$. In that case, there must exist some model $\mathcal{I}$ of $(\mathcal{D}, \mathcal{R})$ with $\mathcal{I} \not\models (\mathfrak{q}, \Xi)$. Note that such "countermodels" can be characterized by the MSO formula $\bigwedge \mathcal{D} \wedge \bigwedge \mathcal{R} \wedge \neg \Xi$. Moreover, any universal model $\mathcal{I}^*$ of $(\mathcal{D}, \mathcal{R})$ must satisfy $\mathcal{I}^* \not\models (\mathfrak{q}, \Xi)$, which can be shown (by contradiction) as follows: Let $\mathcal{I}^*$ be a universal model of $(\mathcal{D}, \mathcal{R})$ and suppose $\mathcal{I}^* \models (\mathfrak{q}, \Xi)$, i.e., $\mathcal{I}^* \models \mathfrak{q}$. Since satisfaction of datalog queries is preserved under homomorphisms and $\mathcal{I}^*$ is universal, we know there exists a homomorphism from $\mathcal{I}^*$ to $\mathcal{I}$, implying $\mathcal{I} \models \mathfrak{q}$, and thus $\mathcal{I} \models (\mathfrak{q}, \Xi)$, contradicting our assumption. As $\mathcal{R}$ is a finite-w set, there exists a universal model $\mathcal{I}^*$ of $(\mathcal{D}, \mathcal{R})$ for which $\mathsf{w}(\mathcal{I}^*)$ is finite. Hence, the following procedure will terminate, witnessing non-entailment: enumerate all natural numbers in their natural order and for each number n initiate a parallel thread with the algorithm from Definition 3 with input n and $\bigwedge \mathcal{D} \wedge \bigwedge \mathcal{R} \wedge \neg \Xi$ (the algorithm is guaranteed to exist due to MSO-friendliness of w). Terminate as soon as one thread does. ◀

4 Cliquewidth and its Properties

In this section, we will propose a width measure, for which we use the term *cliquewidth*. Our definition of this measure works for arbitrary countable instances and thus properly generalizes Courcelle's earlier eponymous notions for finite directed edge-labelled graphs [13] and countable unlabelled undirected graphs [12], as well as Grohe and Turán's cliquewidth notion for finite instances of arbitrary arity [21].

4.1 Cliquewidth of Countable Instances

Intuitively, the notion of cliquewidth is based on the idea of assembling the considered structure (e.g., instance or graph) from its singleton elements (e.g., terms or nodes). To better distinguish these elements during assembly, each may be assigned an initial color (from some finite set $\mathbb{L}$). The "assembly process" consists of successively applying the following operations to previously assembled node-colored structures:

- take the disjoint union of two structures ($\oplus$),
- uniformly assign the color k' to all hitherto k-colored elements ($\mathsf{Recolor}_{k \to k'}$),
- given a predicate $\mathtt{R}$ and a color sequence $\vec{k}$ of length $\mathsf{ar}(\mathtt{R})$, add $\mathtt{R}$-atoms for all tuples of appropriately colored elements ($\mathsf{Add}_{\mathtt{R},\vec{k}}$).

Then, the cliquewidth of a structure is the minimal number of colors needed to assemble it through successive applications of the above operations. For finite structures (e.g., graphs and instances), this is a straightforward, conceivable notion. In order to generalize it to the countably infinite case, one has to find a way to describe infinite assembly processes. In the finite case, an "assembly plan" can be described by an algebraic expression using the above operators, which in turn can be represented by its corresponding "syntax tree." The more elusive idea of an "infinite assembly plan" is then implemented by allowing for infinite, "unfounded" syntax trees. We formalize this idea of "assembly-plan-encoding syntax trees" by representing them as countably infinite instances of a very particular shape.

▶ **Definition 6.** *We define the* infinite binary tree *to be the instance*

$$\mathcal{T}_{\mathrm{bin}} = \{\mathtt{Succ}_0(s, s0) \mid s \in \{0,1\}^*\} \cup \{\mathtt{Succ}_1(s, s1) \mid s \in \{0,1\}^*\}$$

with binary predicates $\mathtt{Succ}_0$ *and* $\mathtt{Succ}_1$*. That is, the nulls of* $\mathcal{T}_{\mathrm{bin}}$ *are denoted by finite sequences of* 0 *and* 1*. The* root *of* $\mathcal{T}_{\mathrm{bin}}$ *is the null identified by the empty sequence, denoted* ε*.*

Given a finite set $\mathbb{L}$ *of* colors*, a finite set* $\mathrm{Cnst} \subseteq \mathfrak{C}$ *of constants, and a finite signature* Σ*, the set* $\mathrm{Dec}(\mathbb{L}, \mathrm{Cnst}, \Sigma)$ *of* decorators *consists of the following unary predicate symbols:*

- $\mathtt{c}_k$ *for any* $\mathtt{c} \in \mathrm{Cnst} \cup \{*\}$ *and* $k \in \mathbb{L}$*,*
- $\mathtt{Add}_{\mathtt{R},\vec{k}}$ *for any* $\mathtt{R} \in \Sigma$ *and* $\vec{k} \in \mathbb{L}^{\mathsf{ar}(\mathtt{R})}$*,*
- $\mathtt{Recolor}_{k \to k'}$ *for* $k, k' \in \mathbb{L}$*,*
- $\oplus$*, and* $\mathtt{Void}$*.*

A $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$-decorated infinite binary tree *(or,* decorated tree *for short) is* $\mathcal{T}_{\mathrm{bin}}$ *extended with facts over* $\mathrm{Dec}(\mathbb{L}, \mathrm{Cnst}, \Sigma)$ *that only use nulls from the original domain of* $\mathcal{T}_{\mathrm{bin}}$*, i.e., from* $\{0,1\}^*$*. A decorated tree* $\mathcal{T}$ *is called a* well-decorated tree *iff*

- *for every null* $s \in \{0,1\}^*$*,* $\mathcal{T}$ *contains exactly one fact* $\mathrm{Dec}(s)$ *with* $\mathrm{Dec} \in \mathrm{Dec}(\mathbb{L}, \mathrm{Cnst}, \Sigma)$*,*
- *for every* $\mathtt{c} \in \mathrm{Cnst}$*,* $\mathcal{T}$ *contains at most one fact of the form* $\mathtt{c}_k(s)$*,*
- *if* $\mathtt{Add}_{\mathtt{R},\vec{k}}(s) \in \mathcal{T}$ *or* $\mathtt{Recolor}_{k \to k'}(s) \in \mathcal{T}$*, then* $\mathtt{Void}(s0) \notin \mathcal{T}$ *and* $\mathtt{Void}(s1) \in \mathcal{T}$*,*
- *if* $\oplus(s) \in \mathcal{T}$*, then* $\mathtt{Void}(s0), \mathtt{Void}(s1) \notin \mathcal{T}$*,*
- *if* $\mathtt{Void}(s) \in \mathcal{T}$ *or* $\mathtt{c}_k(s) \in \mathcal{T}$*, then* $\mathtt{Void}(s0), \mathtt{Void}(s1) \in \mathcal{T}$*.* ⌟

Recall that, due to Rabin's famous Tree Theorem [26], the validity of a given MSO sentence Ξ in $\mathcal{T}_{\mathrm{bin}}$ is decidable. Also, it should be obvious that, given a decorated tree, checking well-decoratedness can be done in first-order logic. More precisely, fixing $\mathbb{L}$, Σ, and Cnst, there is a first-order sentence Φ_{well} such that for any $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$-decorated tree $\mathcal{T}$, $\mathcal{T}$ is well-decorated *iff* $\mathcal{T} \models \Phi_{\mathrm{well}}$.

▶ **Definition 7.** *Let* $\mathcal{T}$ *be a* $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$*-well-decorated tree. We define the function* $\mathsf{ent}^{\mathcal{T}} : \{0,1\}^* \to 2^{\mathrm{Cnst} \cup \{0,1\}^*}$ *mapping each null* $s \in \{0,1\}^*$ *to its* entities *(a set of nulls and constants) as follows:*

$$\mathsf{ent}^{\mathcal{T}}(s) = \{ss' \mid *_k(ss') \in \mathcal{T}, s' \in \{0,1\}^*\} \cup \{\mathtt{c} \mid \mathtt{c}_k(ss') \in \mathcal{T}, \mathtt{c} \in \mathrm{Cnst}, s' \in \{0,1\}^*\}.$$

Every tree node s also endows each of its entities with a color *from $\mathbb{L}$ through the function $\mathsf{col}^{\mathcal{T}}_s : \mathsf{ent}^{\mathcal{T}}(s) \to \mathbb{L}$ in the following way:*

$$\mathsf{col}^{\mathcal{T}}_s(e) = \begin{cases} k & \text{if } e = \mathsf{c} \in \mathrm{Cnst} \text{ and } \mathsf{c}_k(s) \in \mathcal{T}, \text{or if } e = s \in \{0,1\}^* \text{ and } *_k(s) \in \mathcal{T}, \\ k' & \text{if } \mathtt{Recolor}_{k\to k'}(s) \in \mathcal{T} \text{ and } \mathsf{col}^{\mathcal{T}}_{s0}(e) = k, \\ \mathsf{col}^{\mathcal{T}}_{s0}(e) & \text{if } \mathtt{Recolor}_{k\to k'}(s) \in \mathcal{T} \text{ and } \mathsf{col}^{\mathcal{T}}_{s0}(e) \neq k, \text{or if } \mathtt{Add}_{\mathtt{R},\vec{k}}(s) \in \mathcal{T}, \\ \mathsf{col}^{\mathcal{T}}_{sb}(e) & \text{if } \oplus(s) \in \mathcal{T}, e \in \mathsf{ent}^{\mathcal{T}}(sb), \text{ and } b \in \{0,1\}. \end{cases}$$

Every node s is assigned a set of Σ-atoms over its entities as indicated by the sets Atoms_s*:*

$$\mathrm{Atoms}_s = \begin{cases} \{\top(\mathsf{c})\} & \text{if } \mathsf{c}_k(s) \in \mathcal{T}, \mathsf{c} \in \mathrm{Cnst}, \\ \{\top(s)\} & \text{if } *_k(s) \in \mathcal{T}, \\ \{\mathtt{R}(\vec{e}) \mid \mathsf{col}^{\mathcal{T}}_s(\vec{e}) = \vec{k}\} & \text{if } \mathtt{Add}_{\mathtt{R},\vec{k}}(s) \in \mathcal{T}, \\ \emptyset & \text{otherwise.} \end{cases}$$

Defining a colored instance *as a pair $(\mathcal{I}, \lambda)$ of an instance $\mathcal{I}$ and a function λ mapping elements of $\mathsf{dom}(\mathcal{I})$ to colors in a set $\mathbb{L}$, we now associate each node s in $\mathcal{T}$ with the colored Σ-instance $(\mathcal{I}^{\mathcal{T}}_s, \lambda^{\mathcal{T}}_s)$, with $\mathcal{I}^{\mathcal{T}}_s = \bigcup_{s' \in \{0,1\}^*} \mathrm{Atoms}_{ss'}$ and $\lambda^{\mathcal{T}}_s = \mathsf{col}^{\mathcal{T}}_s$. Finally, we define the* colored instance $(\mathcal{I}^{\mathcal{T}}, \lambda^{\mathcal{T}})$ represented by $\mathcal{T}$ *as $(\mathcal{I}^{\mathcal{T}}_\varepsilon, \lambda^{\mathcal{T}}_\varepsilon)$ where ε is the root of $\mathcal{T}$.*

▶ **Definition 8.** *Given a colored instance $(\mathcal{I}, \lambda)$ over a finite set* Cnst *of constants and a countable set of nulls as well as a finite signature Σ, the* cliquewidth *of $(\mathcal{I}, \lambda)$, written $\mathsf{cw}(\mathcal{I}, \lambda)$, is defined to be the smallest natural number n such that $(\mathcal{I}, \lambda)$ is isomorphic to a colored instance represented by some $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$-well-decorated tree with $|\mathbb{L}| = n$. If no such number exists, we let $\mathsf{cw}(\mathcal{I}, \lambda) = \infty$. The cliquewidth $\mathsf{cw}(\mathcal{I})$ of an instance $\mathcal{I}$ is defined to be the minimum cliquewidth over all of its colored versions.* ⌟

▶ **Example 9.** The instance $\mathcal{I}_< = \{\mathtt{R}(n, m) \mid n, m \in \mathbf{N}, n < m\}$ has a cliquewidth of 2, witnessed by the well-decorated tree corresponding to the (non-well-founded) expression E implicitly defined by $E = \mathsf{Add}_{\mathtt{R},1,2}(*_1 \oplus \mathsf{Recolor}_{1\to 2}(E))$.

We will later use the following operation on decorated trees.

▶ **Definition 10.** *Let $(\mathcal{I}, \lambda)$ be a colored instance and $\mathcal{T}$ be a well-decorated tree witnessing that $\mathsf{cw}(\mathcal{I}, \lambda) \leq n$. Then for any $\mathtt{R} \in \Sigma$ we let $\mathsf{Add}_{\mathtt{R},\vec{k}}(\mathcal{I}, \lambda)$ denote the instance $\mathcal{I}^{\mathcal{T}'}$ represented by the well-decorated tree $\mathcal{T}'$ defined as follows:*

- *The root ε of $\mathcal{T}'$ is decorated by $\mathtt{Add}_{\mathtt{R},\vec{k}}$,*
- *the left sub-tree of ε is isomorphic to $\mathcal{T}$,*
- *the right sub-tree of ε is wholly decorated with* `Void`*.* ⌟

4.2 Finite-Cliquewidth Sets and Decidability

We now identify a new class of rule sets for which DaMSOQ query entailment is decidable, that is, the class of *finite-cliquewidth sets.*

▶ **Theorem 11.** *For a fixed $n \in \mathbb{N}$, determining if a given MSO formula Ξ has a model $\mathcal{I}$ with $\mathsf{cw}(\mathcal{I}) \leq n$ is decidable. Thus, cliquewidth is MSO-friendly.*

Proof (Sketch). We use the classical idea of MSO interpretations. Picking $\mathbb{L} = \{1, \ldots, n\}$, one shows that for every given MSO sentence Ξ, one can compute an MSO sentence Ξ', such that for every $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$-well-decorated tree $\mathcal{T}$, $\mathcal{I}^{\mathcal{T}} \models \Xi$ *iff* $\mathcal{T} \models \Xi'$. Thus, checking if Ξ holds in some Cnst, Σ-instance of cliquewidth $\leq n$ can be done by checking if Ξ' holds in some $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$-well-decorated tree, which in turn is equivalent to the existence of a $(\mathbb{L}, \mathrm{Cnst}, \Sigma)$-decorated tree that is a model of $\Xi' \wedge \Phi_{\mathrm{well}}$. Obtain Ξ'' from $\Xi' \wedge \Phi_{\mathrm{well}}$ by reinterpreting all unary predicates as MSO set variables that are quantified over existentially. Ξ'' is an MSO formula over the signature $\{\mathtt{Succ}_0, \mathtt{Succ}_1\}$ which is valid in $\mathcal{T}_{\mathrm{bin}}$ *iff* some decoration exists that makes $\Xi' \wedge \Phi_{\mathrm{well}}$ true. Thus, we have reduced our problem to checking the validity of a MSO sentence in $\mathcal{T}_{\mathrm{bin}}$, which is decidable by Rabin's Tree Theorem [26]. ◀

With this insight in place and the appropriate rule set notion defined, we can leverage Theorem 5 for our decidability result.

▶ **Definition 12.** *A rule set $\mathcal{R}$ is called a* finite-cliquewidth set *(**fcs**) iff for any database $\mathcal{D}$, there exists a universal model for $(\mathcal{D}, \mathcal{R})$ of finite cliquewidth.* ⌟

▶ **Corollary 13.** *For every **fcs** rule set $\mathcal{R}$, the query entailment problem $(\mathcal{D}, \mathcal{R}) \models (\mathtt{q}, \Xi)$ for databases $\mathcal{D}$, and DaMSOQs $(\mathtt{q}, \Xi)$ is decidable.*

In view of Example 9, it is not hard to verify that the rule set $\mathcal{R}^{\infty}_{\mathrm{tran}}$ from the introduction is **fcs**. Yet, it is neither **bts** (as argued before), nor **fus**, which can be observed from the fact that it does not admit a finite rewriting of the BCQ $\mathtt{E}(\mathtt{a}, \mathtt{b})$. Notably, it also does not exhibit *finite controllability* (**fc**), another generic property that guarantees decidability of query entailment [28]. A rule set $\mathcal{R}$ is **fc** *iff* for every $\mathcal{D}$ and CQ q with $(\mathcal{D}, \mathcal{R}) \not\models q$ there exists a *finite* (possibly non-universal) model $\mathcal{I} \models (\mathcal{D}, \mathcal{R})$ with $\mathcal{I} \not\models q$. Picking $\mathcal{D} = \{\mathtt{E}(\mathtt{a}, \mathtt{b})\}$ and $q = \exists x \mathtt{E}(x, x)$ reveals that $\mathcal{R}^{\infty}_{\mathrm{tran}}$ is not **fc**. Therefore, **fcs** encompasses rule sets not captured by any of the popular general decidability classes (namely, **bts**, **fus**, and **fc**). On another note, is no surprise that, akin to **bts**, **fus**, and **fc**, the membership of a rule set in **fcs** is undecidable, which can be argued exactly in the same way as for **bts** and **fus** [3].

4.3 Cliquewidth and Treewidth

We now show that for binary signatures, the class of instances with finite cliquewidth subsumes the class of instances with finite treewidth, implying **bts** $\subseteq$ **fcs**.

▶ **Theorem 14.** *Let $\mathcal{I}$ be a countable instance over a binary signature. If $\mathcal{I}$ has finite treewidth, then $\mathcal{I}$ has finite cliquewidth.*

Proof (Sketch). We convert a tree decomposition T of $\mathcal{I}$ with a width n into a well-decorated tree: (1) By copying nodes, transform T into an infinite binary tree T'. (2) For each term t from $\mathcal{I}$, let the *pivotal* node of t in T' be the node closest to the root containing t. For any two terms t and t' from $\mathcal{I}$ co-occurring in an atom, their pivotal nodes are in an ancestor relationship. (3) Assign one of $n + 1$ "slots" to every term so that in each node of T', every element has a distinct slot. (4) Extract a well-decorated tree from T' by transforming every node into the following bottom-up sequence of operations: (i) $\oplus$-assemble the input from below, (ii) introduce every element for which the current node is pivotal with a color that encodes "open link requests" to elements (identified by their slots) further up, (iii) satisfy the color-link requests from below via $\mathtt{Add}$, (iv) remove the satisfied requests by $\mathtt{Recolor}$. ◀

We note that the converse of Theorem 14 does not hold: Despite its finite cliquewidth, the treewidth of instance $\mathcal{I}_<$ from Example 9 is infinite, as its $\mathtt{R}$-edges form an infinite clique.

To the informed reader, Theorem 14 might not come as a surprise, given that this relationship is known to hold for countable unlabelled undirected graphs[5] [12]. It does, however, cease to hold for infinite structures with predicates of higher arity.

▶ **Example 15.** Let $\mathtt{R}$ be a ternary predicate. The instance $\mathcal{I}_{\text{tern}} = \{\mathtt{R}(-1, n, n{+}1) \mid n \in \mathbb{N}\}$ has a treewidth of 2, however, it does not have finite cliquewidth [15]. Concomitantly, the rule set $\mathcal{R}_{\text{tern}} = \{\mathtt{R}(v, x, y) \to \exists z \mathtt{R}(v, y, z)\}$ is **bts**, but not **fcs**.

While this result may be somewhat discouraging, one can show that its effects can be greatly mitigated by the technique of reification.

▶ **Definition 16.** *Given a finite signature $\Sigma = \Sigma_{\leq 2} \uplus \Sigma_{\geq 3}$ divided into at-most-binary and higher-arity predicates, we define the* reified version *of Σ as the binary signature $\Sigma^{\text{rf}} = \Sigma_{\leq 2} \uplus \Sigma^{\text{rf}}_{2}$ with $\Sigma^{\text{rf}}_{2} = \{\mathtt{R}_i \mid \mathtt{R} \in \Sigma_{\geq 3}, 1 \leq i \leq \mathsf{ar}(\mathtt{R})\}$ a fresh set of binary predicates. The function* reify *maps atoms over $\Sigma_{\leq 2}$ to themselves, while any higher-arity atom $\alpha = \mathtt{R}(t_1, \ldots, t_k)$ with $k \geq 3$ is mapped to the set $\{\mathtt{R}_i(u_\alpha, t_i) \mid 1 \leq i \leq k\}$, where u_α is a fresh null or variable. We lift* reify *to instances, rules, and queries in the natural way.* ⌟

It is best to think of the “reification term” u_α as a locally existentially quantified variable. In particular, in rule heads, u_α will be existentially quantified. Moreover, in datalog queries, reify is only applied to Σ_{EDB}-atoms, while Σ_{IDB}-atoms are left unaltered; this ensures that the result is again a datalog query.

▶ **Example 17.** Consider the instance $\mathcal{I}_{\text{tern}}$ from Example 15. We observe that $\mathsf{reify}(\mathcal{I}_{\text{tern}}) = \{\mathtt{R}_1(u_n, -1), \mathtt{R}_2(u_n, n), \mathtt{R}_3(u_n, n{+}1) \mid n \in \mathbb{N}\}$ has a treewidth of 3 and a cliquewidth of 6, witnessed by the (non-well-founded) expression $\mathsf{Add}_{\mathtt{R}_1,5,1}(*_1 \oplus E)$, where E is implicitly defined via $E = \mathsf{Recolor}_{2\to 3}(\mathsf{Recolor}_{4\to 5}(\mathsf{Recolor}_{3\to 6}(\mathsf{Add}_{\mathtt{R}_3,4,3}(\mathsf{Add}_{\mathtt{R}_2,4,2}(*_2 \oplus (*_4 \oplus E))))))$.

Let us list in all brevity some pleasant and fairly straightforward properties of reification:

(i) If $\mathsf{tw}(\mathcal{I})$ is finite, then so are $\mathsf{tw}(\mathsf{reify}(\mathcal{I}))$ and $\mathsf{cw}(\mathsf{reify}(\mathcal{I}))$.
(ii) $\mathbf{Ch}_\infty(\mathsf{reify}(\mathcal{I}), \mathsf{reify}(\mathcal{R})) \equiv \mathsf{reify}(\mathbf{Ch}_\infty(\mathcal{I}, \mathcal{R}))$.
(iii) If $\mathcal{R}$ is **bts**, then $\mathsf{reify}(\mathcal{R})$ is **bts** and **fcs**.
(iv) $(\mathcal{D}, \mathcal{R}) \models (\mathsf{q}, \Xi)$ *iff* $(\mathsf{reify}(\mathcal{D}), \mathsf{reify}(\mathcal{R})) \models (\mathsf{reify}(\mathsf{q}), \mathsf{reify}(\Xi))$.

▶ **Example 18.** Revisiting Example 15, we can confirm that $\mathsf{reify}(\mathcal{R}_{\text{tern}})$ comprising the rule $\mathtt{R}_1(u, v) \wedge \mathtt{R}_2(u, x) \wedge \mathtt{R}_3(u, y) \to \exists u' z \big(\mathtt{R}_1(u', v) \wedge \mathtt{R}_2(u', y) \wedge \mathtt{R}_3(u', z)\big)$ is **bts** and **fcs**.

The above insights regarding reification allow us to effortlessly reduce any query entailment problem over an arbitrary **bts** rule set to a reasoning problem over a binary **fcs** one. Also, reification is a highly local transformation; it can be performed independently and atom-by-atom on $\mathcal{D}$, $\mathcal{R}$, and (q, Ξ). Therefore, restricting ourselves to **fcs** rule sets does not deprive us of the expressiveness, versatility, and reasoning capabilities offered by arbitrary **bts** rule sets over arbitrary signatures, which includes the numerous classes based on guardedness: guarded and frontier-guarded rules as well as their respective variants weakly, jointly, and glut-(frontier-)guarded rules [3, 7, 23]. It is noteworthy that our line of argument also gives rise to an independent proof of decidability of query entailment for **bts**:[6]

[5] This follows as a direct consequence of a compactness property relating the cliquewidth of countable undirected graphs to that of their finite induced subgraphs.

[6] Note that this result establishes entailment for arbitrary DaMSOQs, while previously reported results [3, 7] only covered CQs. But even when restricting the attention to CQ entailment, the proofs given in these (mutually inspired) prior works do not appear entirely conclusive to us: both invoke a result by Courcelle [11], which, as stated in the title of the article and confirmed by closer inspection, only deals with classes of *finite* structures/graphs with a uniform treewidth bound. Hence, the case of infinite structures is not covered, despite being the prevalent one for universal models. Personal communication with Courcelle confirmed that the case of arbitrary countable structures – although generally believed to hold true – is not an immediate consequence of his result.

▶ **Theorem 19.** *For every* **bts** *rule set $\mathcal{R}$, the query entailment problem $(\mathcal{D}, \mathcal{R}) \models (\mathfrak{q}, \Xi)$ for databases $\mathcal{D}$, and DaMSOQs $(\mathfrak{q}, \Xi)$ is decidable.*

Proof. As argued, $(\mathcal{D}, \mathcal{R}) \models (\mathfrak{q}, \Xi)$ reduces to $(\mathsf{reify}(\mathcal{D}), \mathsf{reify}(\mathcal{R})) \models (\mathsf{reify}(\mathfrak{q}), \mathsf{reify}(\Xi))$. As $\mathcal{R}$ is **bts**, so is $\mathsf{reify}(\mathcal{R})$. The latter being binary, we conclude that it is **fcs**. Then, the claim follows from decidability of DaMSOQ entailment for **fcs** rule sets (Corollary 13). ◀

5 Comparing FCS and FUS

We have seen that the well-known **bts** class is subsumed by the **fcs** class (directly for arities ≤2, and via reification otherwise). As discussed in the introduction, another prominent class of rule sets (incomparable to **bts**) with decidable CQ entailment is the **fus** class. We dedicate the remainder of the paper to mapping out the relationship between **fcs** and **fus**, obtaining the following two results (established in Section 5.1 and Section 5.2, respectively):

▶ **Theorem 20.** *Any* **fus** *rule set of single-headed rules over a binary signature is* **fcs**.

▶ **Theorem 21.** *There exists a* **fus** *rule set of multi-headed rules over a binary signature that is not* **fcs**.

As a consequence of these findings, the necessary restriction to single-headed rules prevents us from wielding the powers of reification in this setting.

5.1 The Case of Single-Headed Rules

In this section, we establish Theorem 20. To this end, let a binary signature Σ, a finite unification set $\mathcal{R}$ of single-headed rules over Σ, and a database $\mathcal{D}$ over Σ be arbitrary but fixed for the remainder of the section. We will abbreviate $\mathbf{Ch}_\infty(\mathcal{D}, \mathcal{R})$ by $\mathbf{Ch}_\infty$. Noting that $\mathbf{Ch}_\infty$ is a universal model, Theorem 20 is an immediate consequence of the following lemma, which we are going to establish in this section.

▶ **Lemma 22.** $\mathbf{Ch}_\infty$ *has finite cliquewidth.*

Looking past datalog. Let $\mathbf{Ch}_\exists \subseteq \mathbf{Ch}_\infty$ be the instance containing the *existential atoms* of $\mathbf{Ch}_\infty$, that is, the atoms derived via the non-datalog rules of $\mathcal{R}$. We show that $\mathbf{Ch}_\exists$ forms a *typed polyforest*, meaning that $\mathbf{Ch}_\exists$ can be viewed as a directed graph where (i) edges are typed with binary predicates from Σ and (ii) when disregarding the orientation of the edges, the graph forms a forest. This implies that $\mathbf{Ch}_\exists$ has treewidth 1. A tree decomposition of $\mathbf{Ch}_\exists$ can be extended into a finite-width tree decomposition of $\mathcal{D} \cup \mathbf{Ch}_\exists$ by adding the finite set $\mathsf{dom}(\mathcal{D})$ to every node. In the sequel, we use $\mathcal{D} \cup \mathbf{Ch}_\exists$ as a basis, to which, in a very controlled manner, we then add the "missing" non-existential atoms derived via datalog rules. Thereby, $\mathcal{R}$ being **fus** will be of great help.

Rewriting datalog rules. We transform the datalog subset of $\mathcal{R}$ into a new rule set $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$, which we then use to fix a set of colors and establish the finiteness of $\mathsf{cw}(\mathbf{Ch}_\infty)$. Letting $\mathcal{R}_{\mathrm{DL}}$ denote the set of all datalog rules from $\mathcal{R}$, we note the following useful equation: $\mathbf{Ch}_\infty = \mathbf{Ch}_\infty(\mathcal{D} \cup \mathbf{Ch}_\exists, \mathcal{R}_{\mathrm{DL}})$ (†). For any $\mathtt{R} \in \Sigma$, we let $\mathsf{rew}(\mathtt{R})$ denote the datalog rule set $\{\varphi(\vec{x}, \vec{y}) \to \mathtt{R}(\vec{y}) \mid \exists \vec{x} \varphi(\vec{x}, \vec{y}) \in \mathsf{rew}_{\mathcal{R}}(\mathtt{R}(\vec{y}))\}$ giving rise to the overall datalog rule set $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}} = \bigcup_{\mathtt{R} \in \Sigma} \mathsf{rew}(\mathtt{R})$. We now show that the rule set $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$ admits an important property:

▶ **Lemma 23.** *For any $\mathtt{R}(\vec{t}) \in \mathbf{Ch}_\infty \setminus (\mathcal{D} \cup \mathbf{Ch}_\exists)$, there exists a trigger (ρ, h) in $\mathcal{D} \cup \mathbf{Ch}_\exists$ with $\rho \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$ such that applying (ρ, h) adds $\mathtt{R}(\vec{t})$ to $\mathcal{D} \cup \mathbf{Ch}_\exists$.*

Proof. Given that $\mathcal{R}$ is **fus**, it follows that for any $\mathsf{ar}(\mathtt{R})$-tuple $\vec{t}$ of terms from $\mathcal{D} \cup \mathbf{Ch}_{\exists}$, there exists a CQ $\exists\vec{x}\varphi(\vec{x},\vec{y}) \in \mathsf{rew}_{\mathcal{R}}(\mathtt{R}(\vec{y}))$ such that the following holds: $\mathbf{Ch}_{\infty}(\mathcal{D} \cup \mathbf{Ch}_{\exists}, \mathcal{R}) \models \mathtt{R}(\vec{t})$ *iff* $\mathcal{D} \cup \mathbf{Ch}_{\exists} \models \exists\vec{x}\varphi(\vec{x},\vec{t})$. Thus, said trigger exists for some $\rho \in \mathsf{rew}(\mathtt{R})$ in $\mathcal{D} \cup \mathbf{Ch}_{\exists}$. ◀

From Lemma 23 and (†), we conclude $\mathbf{Ch}_1(\mathcal{D} \cup \mathbf{Ch}_{\exists}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}) = \mathbf{Ch}_{\infty}$ (‡). This tells us that we can apply rules of $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$ in one step to obtain $\mathbf{Ch}_{\infty}$. Ultimately, we will leverage this to bound the cliquewidth of $\mathbf{Ch}_{\infty}$. In view of (‡), we may reformulate Lemma 22 as follows:

▶ **Lemma 24.** $\mathbf{Ch}_1(\mathcal{D} \cup \mathbf{Ch}_{\exists}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}})$ *has finite cliquewidth.*

Separating connected from disconnected rules. Next, we distinguish between two types of rules in $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$. We say a datalog rule is *disconnected iff* its head variables belong to distinct connected components in its body; otherwise it is called *connected*. We let $\mathcal{R}_{\mathrm{DL}}^{\mathrm{conn}}$ denote all connected rules of $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$ and let $\mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$ denote all disconnected rules. One can observe that upon applying connected rules, frontier variables can only be mapped to – and hence connect – “nearby terms” of $\mathcal{D} \cup \mathbf{Ch}_{\exists}$ (using a path-based distance, bounded by the size of rule bodies), which, together with our insights about the structure of $\mathcal{D} \cup \mathbf{Ch}_{\exists}$, permits the construction of a tree decomposition of finite width, giving rise to the following lemma:

▶ **Lemma 25.** $\mathbf{Ch}_1(\mathcal{D} \cup \mathbf{Ch}_{\exists}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{conn}})$ *has finite treewidth.*

Note that this lemma does not generalize to all of $\mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}$, since disconnected rules from $\mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$ might realize “concept products” as in $\mathtt{A}(x) \wedge \mathtt{A}(y) \rightarrow \mathtt{E}(x,y)$. Clearly, such rules from $\mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$ are the reason why **bts** fails to subsume **fus**. Let us define $\mathbf{Ch}_{\exists+} = \mathbf{Ch}_1(\mathcal{D} \cup \mathbf{Ch}_{\exists}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{conn}})$. As $\mathbf{Ch}_1(\mathbf{Ch}_{\exists}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{rew}}) = \mathbf{Ch}_1(\mathbf{Ch}_{\exists+}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}})$, all that is left to show is:

▶ **Lemma 26.** $\mathbf{Ch}_1(\mathbf{Ch}_{\exists+}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}})$ *has finite cliquewidth.*

Searching for a suitable coloring. Having established the finite treewidth of $\mathbf{Ch}_{\exists+}$ by Lemma 25, its finite cliquewidth can be inferred by Theorem 14, implying the existence of a well-decorated tree $\mathcal{T}$ with $\mathcal{I}^{\mathcal{T}} = \mathbf{Ch}_{\exists+}$. Moreover, we know that $\mathbf{Ch}_{\exists+}$ already contains all terms of $\mathbf{Ch}_{\infty}$. Thus, all that remains is to “add” the missing datalog atoms of $\mathbf{Ch}_{\infty} \setminus \mathbf{Ch}_{\exists+}$ to $\mathcal{I}^{\mathcal{T}}$. Certainly, the coloring function $\mathsf{col}_{\varepsilon}^{\mathcal{T}}$ provided by $\mathcal{T}$ cannot be expected to be very helpful in this task. To work around this, the following technical lemma ensures that an arbitrary coloring can be installed on top of a given instance of finite cliquewidth.

▶ **Lemma 27** (Recoloring Lemma). *Let $\mathcal{I}$ be an instance satisfying* $\mathsf{cw}(\mathcal{I}) = n$ *and let* $\lambda' : \mathsf{dom}(\mathcal{I}) \rightarrow \mathbb{L}'$ *be an arbitrary coloring of $\mathcal{I}$. Then* $\mathsf{cw}(\mathcal{I}, \lambda') \leq (n+1) \cdot |\mathbb{L}'|$.

Proof (Sketch). Let $\mathsf{cw}(\mathcal{I}) = n$ be witnessed by a well-decorated tree $\mathcal{T}$. Let $\mathbb{L}$ with $|\mathbb{L}| = n$ be the set of colors used in $\mathcal{T}$. We construct a well-decorated tree $\mathcal{T}^{\lambda'}$ representing $(\mathcal{I}, \lambda')$ and using the color set $(\mathbb{L} \times \mathbb{L}') \uplus \mathbb{L}'$, thus witnessing $\mathsf{cw}(\mathcal{I}, \lambda') \leq (n+1) \cdot |\mathbb{L}'|$. We obtain $\mathcal{T}^{\lambda'}$ from $\mathcal{T}$ by modifying each node s of $\mathcal{T}$ as follows:

- If s is labeled with $*_k$, we change it to $*_{(k,\lambda'(s))}$.
- If s is labeled with $\mathtt{c}_k$ with $\mathtt{c} \in \mathrm{Cnst}$, we change it to $\mathtt{c}_{(k,\lambda'(\mathtt{c}))}$.
- If s is labeled with $\mathtt{Add}_{\mathtt{R},k}$, we replace s with a sequence of nodes, one for each decorator in $\{\mathtt{Add}_{\mathtt{R},(k,\ell)} \mid \ell \in \mathbb{L}'\}$. We proceed in an analogous fashion for decorators $\mathtt{Add}_{\mathtt{R},k,k'}$ and $\mathtt{Recolor}_{k \rightarrow k'}$.
- If v is labeled with other decorators, we keep it as is.

Last, on top of the obtained tree, we apply a “color projection” to $\mathbb{L}'$ by adding recoloring statements of the form $\mathtt{Recolor}_{(k,\ell) \rightarrow \ell}$ for all $(k,\ell) \in \mathbb{L} \times \mathbb{L}'$. To complete the construction, we add the missing nodes to $\mathcal{T}^{\lambda'}$, each decorated with $\mathtt{Void}$. ◀

We proceed by defining types for elements in $\mathbf{Ch}_{\exists+}$, giving rise to the desired coloring. For the following, note that by definition, any rule from $\mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$ must have two frontier variables.

▶ **Definition 28.** *Let $\rho \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$ with $\mathsf{body}(\rho) = \varphi(x_1, x_2, \vec{y})$, with x_1, x_2 frontier variables. Let us define $\rho_1(x) = \exists \vec{y} x_2\ \varphi(x, x_2, \vec{y})$ and $\rho_2(x) = \exists \vec{y} x_1\ \varphi(x_1, x, \vec{y})$. Then, for a term $t \in \mathsf{dom}(\mathbf{Ch}_{\exists+})$, we define its* type *$\tau(t)$ as $\{\rho_i(x) \mid \rho \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}},\ \mathbf{Ch}_{\exists+} \models \rho_i(t), i \in \{1, 2\}\}$.* ⌟

Now, we define our new coloring function $\lambda_\exists$. Given a term t of $\mathbf{Ch}_{\exists+}$, we let $\lambda_\exists(t) = \tau(t)$.

▶ **Corollary 29.** *There exists an $n_\exists \in \mathbb{N}$ with $\mathsf{cw}(\mathbf{Ch}_{\exists+}, \lambda_\exists) = n_\exists$.*

Proof. From Lemma 25 and Theorem 14, we know that $\mathsf{cw}(\mathbf{Ch}_{\exists+})$ is finite. Hence, there exists a natural number n and a coloring λ such that $\mathsf{cw}(\mathbf{Ch}_{\exists+}, \lambda) = n$. Moreover, the codomain of $\lambda_\exists$ is finite, say n'. Thus, we get $n_\exists = (n + 1) \cdot n'$ by Lemma 27. ◀

Coping with disconnected rules. We now conclude the proof of Lemma 26, i.e., we show that $\mathsf{cw}(\mathbf{Ch}_1(\mathbf{Ch}_{\exists+}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}), \lambda_\exists) \leq n_\exists$, thereby proving Theorem 20. To this end, consider some rule $\rho \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$. As stated earlier, we can assume that ρ is of the form $\varphi(x_1, x_2, \vec{y}) \to \mathtt{R}(x_1, x_2)$. From this, we obtain the following useful correspondence:

▶ **Lemma 30.** *For any $t, t' \in \mathsf{dom}(\mathbf{Ch}_{\exists+})$ and $\rho \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$,*

$$\mathbf{Ch}_{\exists+} \models \exists \vec{y} \varphi(t, t', \vec{y}) \quad \textit{iff} \quad \rho_1(x) \in \tau(t) \textit{ and } \rho_2(x) \in \tau(t').$$

Proof. ($\Rightarrow$) follows from the definition of each set. For ($\Leftarrow$), we exploit disconnectedness and split $\varphi(x_1, x_2, \vec{y})$ into two distinct parts. Let $\varphi_1(x_1, \vec{y}_1)$ be the connected component of $\varphi(x_1, x_2, \vec{y})$ that contains x_1, and $\varphi_2(x_2, \vec{y}_2)$ denote the remainder (which includes the connected component of x_2). By assumption, $\vec{y}_1$ and $\vec{y}_2$ are disjoint, whence $\exists \vec{y} \varphi(t, t', \vec{y})$ is equivalent to $\exists \vec{y}_1 \varphi_1(t, \vec{y}_1) \wedge \exists \vec{y}_2 \varphi_2(t', \vec{y}_2)$. Note that $\rho_1(x) \in \tau(t)$ implies $\mathbf{Ch}_{\exists+} \models \exists \vec{y}_1 \varphi_1(t, \vec{y}_1)$, while $\rho_2(x) \in \tau(t')$ implies $\mathbf{Ch}_{\exists+} \models \exists \vec{y}_2 \varphi_2(t', \vec{y}_2)$. Therefore, $\mathbf{Ch}_{\exists+} \models \exists \vec{y} \varphi(t, t', \vec{y})$. ◀

▶ **Lemma 31.** *Let $\rho \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$ be of the form $\varphi(x_1, x_2, \vec{y}) \to \mathtt{R}(x_1, x_2)$. Then,*

$$\bigcup_{\rho_1(x) \in \ell,\ \rho_2(x) \in \ell'} \mathsf{Add}_{\mathtt{R}, \ell, \ell'}(\mathbf{Ch}_{\exists+}, \lambda_\exists) = \mathbf{Ch}_{\exists+} \cup \{\mathtt{R}(t, t') \mid \mathbf{Ch}_{\exists+} \models \exists \vec{y} \varphi(t, t', \vec{y})\}.$$

Proof. All $\mathbf{Ch}_{\exists+}$ atoms are contained in both sides of the equation. Considering any $\mathtt{R}(t, t') \notin \mathbf{Ch}_{\exists+}$ we find it contained in the left-hand side *iff* $\rho_1(x) \in \lambda_\exists(t) = \tau(t)$ and $\rho_2(x) \in \lambda_\exists(t') = \tau(t')$ *iff* $\mathbf{Ch}_{\exists+} \models \exists \vec{y} \varphi(t, t' \vec{y})$ (by Lemma 30) *iff* $\mathtt{R}(t, t')$ is contained in the right-hand side of the equation. ◀

Of course, the right hand side of the equation in Lemma 31 coincides with $\mathbf{Ch}_1(\mathbf{Ch}_{\exists+}, \{\rho\})$. We observe – given that the coloring $\lambda_\exists$ remains unaltered – that the finitely many distinct applications of $\mathsf{Add}_{\mathtt{R}, \ell, \ell'}$ in Lemma 31 are independent and can be chained without changing the result. Further, no such application increases the cliquewidth of the instance it is applied to. Since these arguments can be lifted to the application of *all* rules from $\mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}$, we get

$$\mathsf{cw}\Big(\mathbf{Ch}_{\exists+} \cup \bigcup_{\varphi(x_1, x_2, \vec{y}) \to \mathtt{R}(x_1, x_2) \in \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}}} \{\mathtt{R}(t, t') \mid \mathbf{Ch}_{\exists+} \models \exists \vec{y} \varphi(t, t', \vec{y})\}\Big) \leq n_\exists.$$

Observe that the considered instance is equal to $\mathbf{Ch}_1(\mathbf{Ch}_{\exists+}, \mathcal{R}_{\mathrm{DL}}^{\mathrm{disc}})$. Hence, we have established Lemma 26, concluding Lemma 24, which finishes the proof of Lemma 22, thus entailing the desired Theorem 20.

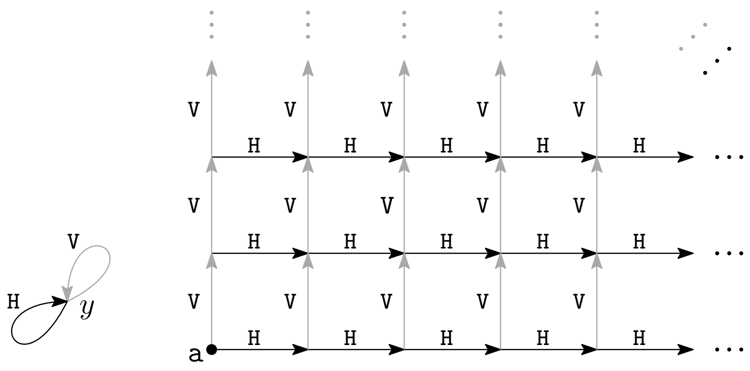

Figure 1 The instance $\mathcal{G}_\infty$.

5.2 The Case of Multi-Headed Rules

We will now prove Theorem 21, which implies that for multi-headed rules **fus** $\not\subseteq$ **fcs**. To this end, we exhibit a **fus** rule set that yields universal models of infinite cliquewidth. Let $\Sigma_{\text{grid}} = \{\mathtt{H}, \mathtt{V}\}$ with $\mathtt{H}, \mathtt{V}$ binary predicates. Let $\mathcal{R}_{\text{grid}}$ denote the following rule set over Σ_{grid}:

$$
\begin{array}{lrl}
\text{(loop)} & & \to \exists x \ \big(\mathtt{H}(x, x) \wedge \mathtt{V}(x, x)\big) \\
\text{(grow)} & & \to \exists y y' \big(\mathtt{H}(x, y) \wedge \mathtt{V}(x, y')\big) \\
\text{(grid)} & \mathtt{H}(x, y) \wedge \mathtt{V}(x, x') & \to \exists y' \ \big(\mathtt{H}(x', y') \wedge \mathtt{V}(y, y')\big)
\end{array}
$$

We make use of $\mathcal{R}_{\text{grid}}$ to establish Theorem 21, the proof of which consists of two parts. First, we provide a database $\mathcal{D}_{\text{grid}}$ to form a knowledge base $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$ for which no universal model of finite cliquewidth exists (Lemma 33). Second, we show that $\mathcal{R}_{\text{grid}}$ is a finite unification set (Lemma 34).

$\mathcal{R}_{\text{grid}}$ is not a finite-cliquewidth set. Toward establishing this result, let $\mathcal{D}_{\text{grid}} = \{\top(\mathtt{a})\}$ and define the instance $\mathcal{G}_\infty$, where $\mathtt{a}$ is a constant and y as well as $x_{i,j}$ with $i, j \in \mathbb{N}$ are nulls:

$$
\begin{array}{rcl}
\mathcal{G}_\infty & = & \big\{\mathtt{H}(\mathtt{a}, x_{1,0}), \mathtt{V}(\mathtt{a}, x_{0,1}), \mathtt{H}(y, y), \mathtt{V}(y, y)\big\} \\
& & \cup \big\{\mathtt{H}(x_{i,j}, x_{i+1,j}), \mathtt{V}(x_{i,j}, x_{i,j+1}) \mid (i, j) \in (\mathbb{N} \times \mathbb{N}) \setminus \{(0, 0)\}\big\}.
\end{array}
$$

Figure 1 depicts $\mathcal{G}_\infty$ graphically. The following lemma summarizes consecutively established properties of $\mathcal{G}_\infty$ and its relationship with $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$.

▶ **Lemma 32.** *With $\mathcal{G}_\infty$ and $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$ as given above, we obtain:*

- *$\mathcal{G}_\infty$ has infinite cliquewidth,*
- *$\mathcal{G}_\infty$ is a universal model of $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$,*
- *the only homomorphism $h : \mathcal{G}_\infty \to \mathcal{G}_\infty$ is the identity,*
- *any universal model of $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$ contains an induced sub-instance isomorphic to $\mathcal{G}_\infty$,*
- *$(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$ has no universal model of finite cliquewidth.*

From the last point (established via the insight that taking induced subinstances never increases cliquewidth), the announced result immediately follows.

▶ **Lemma 33.** *$\mathcal{R}_{\text{grid}}$ is not* **fcs**.

$\mathcal{R}_{\mathbf{grid}}$ is a finite-unification set. Unfortunately, $\mathcal{R}_{\text{grid}}$ does not fall into any of the known syntactic **fus** subclasses and showing that a provided rule set is **fus** is a notoriously difficult task in general.[7] At least, thanks to the rule (loop), every BCQ that is free of constants is always entailed and can thus be trivially re-written. For queries involving constants, we provide a hand-tailored query rewriting algorithm, along the lines of prior work exploring the multifariousness of **fus** [25]. The required argument is quite elaborate and we just sketch the main ideas here due to space restrictions.

We make use of special queries, referred to as *marked queries*: queries with some of their terms "marked" with the purpose of indicating terms that need to be mapped to database constants in the course of rewriting. The notion of query satisfaction is lifted to such marked CQs, which enables us to identify the subclass of *properly marked queries* as those who actually have a match into some instance $\mathbf{Ch}_\infty(\mathcal{D}, \mathcal{R}_{\text{grid}})$, where $\mathcal{D}$ is an arbitrary database.

With these notions at hand, we can now define a principled rewriting procedure consisting of the exhaustive application of three different types of transformation rules. We show that this given set of transformations is in fact sound and complete, i.e., it produces correct first-order rewritings upon termination. Last, we provide a termination argument by showing that the operations reduce certain features of the rewritten query. Thus, we arrive at the second announced result, completing the overall proof.

▶ **Lemma 34.** *$\mathcal{R}_{\text{grid}}$ is* **fus***.*

5.3 Fus and Expressive Queries

Let us put the negative result of Section 5.2 into perspective. Thanks to Corollary 13, we know that **fcs** ensures decidability of arbitrary DaMSOQ entailment, a quite powerful class of queries. By contrast, **fus** is a notion tailored to CQs and unions thereof. As it so happens, searching for a method to establish decidability of DaMSOQ entailment for arbitrary **fus** rule sets turns out to be futile. This even holds for a fixed database and a fixed rule set.

▶ **Lemma 35.** *DaMSOQ entailment is undecidable for $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$.*

The corresponding proof is rather standard: One can, given a deterministic Turing machine TM, create a DaMSOQ $(\mathfrak{q}_{TM}, \Xi_{TM})$ that is even expressible in monadic datalog and, when evaluated over $\mathcal{G}_\infty$, uses the infinite grid to simulate a run of that Turing machine, resulting in a query match *iff* TM halts on the empty tape. As $\mathcal{G}_\infty$ is a universal model of $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}})$, and DaMSOQ satisfaction is preserved under homomorphisms, the entailment $(\mathcal{D}_{\text{grid}}, \mathcal{R}_{\text{grid}}) \models (\mathfrak{q}_{TM}, \Xi_{TM})$ coincides with the termination of TM, concluding the argument.

6 Conclusions and Future Work

In this paper, we have introduced a generic framework, by means of which model-theoretic properties of a class of existential rule sets can be harnessed to establish that the entailment of *datalog/MSO queries* – a very expressive query formalism subsuming numerous popular query languages – is decidable for that class. We have put this framework to use by introducing *finite-cliquewidth sets* and clarified this class's relationship with notable others, resulting in the insight that a plethora of known as well as hitherto unknown decidability results can be uniformly obtained via the decidability of DaMSOQ entailment over finite-cliquewidth sets of rules. Our results entail various appealing directions for follow-up investigations:

[7] This should not be too surprising however, as being **fus** is an undecidable property [3].

- Certainly, the class of single-headed binary **fus** rule sets is not the most general fragment of **fus** subsumed by **fcs**. We strongly conjecture that many of the known, syntactically defined, "concrete subclasses" of **fus** are actually contained in **fcs**, and we are confident that corresponding results can be established.
- Conversely, we are striving to put the notion of **fcs** to good use by identifying comprehensive, syntactically defined subclasses of **fcs**, enabling decidable, highly expressive querying beyond the realms of **bts**, **fus**, or **fc**. As a case in point, the popular modeling feature of *transitivity* of a relation has been difficult to accommodate in existing frameworks [2, 17, 22, 32], whereas the observations presented in this paper seem to indicate that **fcs** can tolerate transitivity rules well and natively.
- Finally, we are searching for even more general MSO-friendly width notions that give rise to classes of rule sets subsuming **fcs**, and which also encompass **bts** natively, without arity restrictions or "reification detours."

Aside from these major avenues for future research, there are also interesting side roads worth exploring:

- Are there other, more general model-theoretic criteria of "structural well-behavedness" that, when ensured for universal models, guarantee decidability of query entailment "just" for (U)CQs? Clearly, if we wanted **fus** to be subsumed by such a criterion, it would have to accept structures like $\mathcal{G}_\infty$ (i.e., infinite grids) and we would have to relinquish decidability of DaMSOQ entailment.
- More generally: Are there other "decidability sweet spots" between expressibility of query classes and structural restrictions on universal models?

References

1 Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995. URL: `http://webdam.inria.fr/Alice/`.

2 Jean-François Baget, Meghyn Bienvenu, Marie-Laure Mugnier, and Swan Rocher. Combining existential rules and transitivity: Next steps. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, pages 2720–2726. AAAI Press, 2015. URL: `http://ijcai.org/Abstract/15/385`.

3 Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat. On rules with existential variables: Walking the decidability line. *Artificial Intelligence*, 175(9):1620–1654, 2011. `doi:10.1016/j.artint.2011.03.002`.

4 Catriel Beeri and Moshe Y. Vardi. A proof procedure for data dependencies. *Journal of the ACM*, 31(4):718–741, 1984. `doi:10.1145/1634.1636`.

5 Pierre Bourhis, Markus Krötzsch, and Sebastian Rudolph. How to best nest regular path queries. In Meghyn Bienvenu, Magdalena Ortiz, Riccardo Rosati, and Mantas Simkus, editors, *Informal Proceedings of the 27th International Workshop on Description Logics, Vienna, Austria, July 17-20, 2014*, volume 1193 of *CEUR Workshop Proceedings*, pages 404–415. CEUR-WS.org, 2014. URL: `http://ceur-ws.org/Vol-1193/paper_80.pdf`.

6 Pierre Bourhis, Michael Morak, and Andreas Pieris. Making cross products and guarded ontology languages compatible. In Carles Sierra, editor, *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, pages 880–886. ijcai.org, 2017. `doi:10.24963/ijcai.2017/122`.

7 Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Artificial Intelligence Research*, 48:115–174, 2013. `doi:10.1613/jair.3873`.

8 Andrea Calì, Georg Gottlob, and Andreas Pieris. Towards more expressive ontology languages: The query answering problem. *Artificial Intelligence*, 193:87–128, 2012. `doi:10.1016/j.artint.2012.08.002`.

9 Ashok K. Chandra, Harry R. Lewis, and Johann A. Makowsky. Embedded implicational dependencies and their inference problem. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing (STOC'81)*, pages 342–354. ACM, 1981. `doi:10.1145/800076.802488`.

10 Stavros S. Cosmadakis, Haim Gaifman, Paris C. Kanellakis, and Moshe Y. Vardi. Decidable optimization problems for database logic programs (preliminary report). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC'88)*, pages 477–490. ACM, 1988. `doi:10.1145/62212.62259`.

11 Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. `doi:10.1016/0890-5401(90)90043-H`.

12 Bruno Courcelle. Clique-width of countable graphs: A compactness property. *Discrete Mathematics*, 276(1-3):127–148, 2004. `doi:10.1016/S0012-365X(03)00303-0`.

13 Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic – A Language-Theoretic Approach*, volume 138 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 2012. URL: `http://www.cambridge.org/fr/knowledge/isbn/item5758776/?site_locale=fr_FR`.

14 Alin Deutsch, Alan Nash, and Jeffrey B. Remmel. The chase revisited. In Maurizio Lenzerini and Domenico Lembo, editors, *Proceedings of the 27th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'08)*, pages 149–158. ACM, 2008. `doi:10.1145/1376916.1376938`.

15 Thomas Feller, Tim S. Lyon, Piotr Ostropolski-Nalewaja, and Sebastian Rudolph. Finite-cliquewidth sets of existential rules: Toward a general criterion for decidable yet highly expressive querying. *CoRR*, abs/2209.02464, 2022. `doi:10.48550/arXiv.2209.02464`.

16 Daniela Florescu, Alon Y. Levy, and Dan Suciu. Query containment for conjunctive queries with regular expressions. In Alberto O. Mendelzon and Jan Paredaens, editors, *Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'98)*, pages 139–148. ACM, 1998. `doi:10.1145/275487.275503`.

17 Harald Ganzinger, Christoph Meyer, and Margus Veanes. The two-variable guarded fragment with transitive relations. In *Proceedings of the 14th Annual IEEE Symposium on Logic in Computer Science (LICS'99)*, pages 24–34. IEEE Computer Society, 1999. `doi:10.1109/LICS.1999.782582`.

18 Birte Glimm, Carsten Lutz, Ian Horrocks, and Ulrike Sattler. Conjunctive query answering for the description logic SHIQ. *Journal of Artificial Intelligence Research*, 31:157–204, 2008. `doi:10.1613/jair.2372`.

19 Kurt Gödel. *Über die Vollständigkeit des Logikkalküls*. PhD thesis, Universität Wien, 1929.

20 Georg Gottlob. Datalog+/-: A unified approach to ontologies and integrity constraints. In Valeria De Antonellis, Silvana Castano, Barbara Catania, and Giovanna Guerrini, editors, *Proceedings of the 17th Italian Symposium on Advanced Database Systems, (SEBD'09)*, pages 5–6. Edizioni Seneca, 2009.

21 Martin Grohe and György Turán. Learnability and definability in trees and similar structures. *Theory of Computing Systems*, 37(1):193–220, 2004. `doi:10.1007/s00224-003-1112-8`.

22 Emanuel Kieronski and Sebastian Rudolph. Finite model theory of the triguarded fragment and related logics. In *Proceedings of the 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'21)*, pages 1–13. IEEE, 2021. `doi:10.1109/LICS52264.2021.9470734`.

23 Markus Krötzsch and Sebastian Rudolph. Extending decidable existential rules by joining acyclicity and guardedness. In Toby Walsh, editor, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*, pages 963–968. IJCAI/AAAI, 2011. `doi:10.5591/978-1-57735-516-8/IJCAI11-166`.

24 Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Query answering in the horn fragments of the description logics SHOIQ and SROIQ. In Toby Walsh, editor, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*, pages 1039–1044. IJCAI/AAAI, 2011. `doi:10.5591/978-1-57735-516-8/IJCAI11-178`.

25 Piotr Ostropolski-Nalewaja, Jerzy Marcinkowski, David Carral, and Sebastian Rudolph. A journey to the frontiers of query rewritability. *CoRR*, abs/2012.11269, 2020. arXiv:2012.11269.

26 Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969. URL: http://www.jstor.org/stable/1995086.

27 Juan L. Reutter, Miguel Romero, and Moshe Y. Vardi. Regular queries on graph databases. *Theory of Computing Systems*, 61(1):31–83, 2017. doi:10.1007/s00224-016-9676-2.

28 Riccardo Rosati. On the decidability and finite controllability of query processing in databases with incomplete information. In Stijn Vansummeren, editor, *Proceedings of the 25th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'06)*, pages 356–365. ACM, 2006. doi:10.1145/1142351.1142404.

29 Sebastian Rudolph and Markus Krötzsch. Flag & check: Data access with monadically defined queries. In Richard Hull and Wenfei Fan, editors, *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'13)*, pages 151–162. ACM, 2013. doi:10.1145/2463664.2465227.

30 Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. All elephants are bigger than all mice. In Franz Baader, Carsten Lutz, and Boris Motik, editors, *Proceedings of the 21st International Workshop on Description Logics (DL2008)*, volume 353 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008. URL: http://ceur-ws.org/Vol-353/RudolphKraetzschHitzler.pdf.

31 Eric Salvat and Marie-Laure Mugnier. Sound and complete forward and backward chainings of graph rules. In Peter W. Eklund, Gerard Ellis, and Graham Mann, editors, *Proceedings of the 4th International Conference on Conceptual Structures (ICCS'96)*, volume 1115 of *LNCS*, pages 248–262. Springer, 1996. doi:10.1007/3-540-61534-2_16.

32 Wieslaw Szwast and Lidia Tendera. The guarded fragment with transitive guards. *Annals of Pure and Applied Logic*, 128(1-3):227–276, 2004. doi:10.1016/j.apal.2004.01.003.