# Revisiting Acyclicity and Guardedness Criteria for Decidability of Existential Rules

## Technical Report

Markus Krötzsch and Sebastian Rudolph

markus.kroetzsch@comlab.ox.ac.uk
University of Oxford, UK

sebastian.rudolph@kit.edu
KIT, Germany

**Abstract.** Existential rules, i.e. Datalog extended with existential quantifiers in rule heads, are currently studied under a variety of names such as *Datalog+/–*, *∀∃-rules*, and *tuple-generating dependencies*. The renewed interest in this formalism is fuelled by a wealth of recently discovered language fragments for which query answering is decidable. This paper extends and consolidates two of the main approaches in this field – acyclicity and guardedness – by providing (1) complexity-preserving generalisations of weakly acyclic and weakly (frontier-)guarded rules, and (2) a novel formalism of *glut-(frontier-)guarded rules* that subsumes both. This builds on an insight that acyclicity can be used to extend any existential rule language while retaining decidability. Besides decidability, combined query complexities are established in all cases.

## 1 Introduction

Rule-based knowledge representation has a long-standing history in AI and related areas such as databases and information systems. Function-free first-order Horn logic (also referred to as Datalog) as one of the central paradigms, however, has been criticised for its inability of stating or inferring the existence of domain entities not previously introduced as constants [18]. Existential rules, i.e. Datalog extended by *value invention* capabilities realised by existential quantifiers in rule heads, overcome this restriction and are currently studied under a variety of names such as *Datalog+/–*, *∀∃-rules*, and – primarily in the database community – *tuple-generating dependencies* (TGDs) [2, 3, 8, 9, 7, 6, 15, 14]. The recent interest in this formalism marks the convergence of two paradigms of knowledge representation research that used to be rather separated: rule-based approaches and ontology languages.

This new ground was found to be very fertile, as witnessed by the above works' discoveries of many new rule languages for which query answering is decidable. Widely varying data and combined complexities underline the richness of the field. Examples of application areas for this new family of knowledge representation languages range from data exchange and data integration [15] to ontological data access in the spirit of the ontology languages of the DL-Lite family [7, 10]. The wealth of recent contributions

supports the development of such applications, but also calls for a more unified view on the existing proposals, their exact relationships, and formal properties. This is the general incentive for this work.

Concretely, we extend and consolidate two of the main notions commonly employed to ensure decidability: acyclicity and guardedness. The main contributions are as follows.

1. We extend weak acyclicity and weak (frontier-) guardedness to obtain *joint acyclicity* and *joint (frontier-) guardedness*. Both extensions use the observation that the existing notions over-estimate how far values can be passed on within a rule set, and that there is a refined criterion that still can be checked in polynomial time.
2. We present a new method of eliminating existential quantifiers from jointly acyclic rule sets. The approach incurs an exponential blow-up but is still worst-case optimal. The relevance of the method stems from the insight that a partial application of the procedure can also simplify rule sets that are not jointly acyclic.
3. We apply this observation to combine guardedness and acyclicity in the language of *glut-(frontier-)guarded rules*, based on identifying *glut variables* that may represent an overabundance of "existentially invented" values. Only glut variables remain affected by existential quantifiers after applying the elimination method introduced for jointly acyclic rules.

An important insight of this work therefore is that a very general notion of acyclicity can be combined "modularly" with existing rule languages without losing decidability. Jointly frontier-guarded rules serve us as an example for this construction, and illustrate that further studies are needed to determine the exact complexity of reasoning in each case. We determine exact combined worst-case complexities for all rule languages introduced herein.

Section 2 provides the preliminaries and reviews the existing results in the field. We then motivate and introduce the notion of joint acyclicity in Section 3, and present a generic way of eliminating jointly acyclic variables in Section 4. Section 5 introduces jointly frontier-guarded rules, and Section 6 combines all previous ideas to obtain *glut-(frontier-)guarded rules* for which the combined complexity of query answering is shown to be 3ExpTime-complete. Section 7 concludes.

## 2 Existential Rules

We now provide the basic notions of the logical framework we consider, followed by an overview of a number of important approaches in this area.

**Definition 1.** *Consider a* signature $\langle \mathbf{C}, \mathbf{P}, \mathbf{V} \rangle$ *consisting of a finite set of* constant symbols $\mathbf{C}$*, a finite set of* predicates $\mathbf{P}$*, and an infinite set of* variables $\mathbf{V}$*, all of which are mutually disjoint. A function* $\mathsf{ar} : \mathbf{P} \to \mathbb{N}$ *associates a natural number* $\mathsf{ar}(s)$ *with each predicate* $r \in \mathbf{P}$ *that defines the (unique)* arity *of* $r$*. The set of* positions *of a predicate* $r$ *is the set* $\Pi_r = \{\langle r, 1\rangle, \ldots, \langle r, \mathsf{ar}(r)\rangle\}$.

- *A* term *is a variable* $x \in \mathbf{V}$ *or a constant* $c \in \mathbf{C}$.

– *An* atom *is a formula of the form* $r(t_1, \ldots, t_n)$ *if* $t_1, \ldots, t_n$ *are terms, and* $r \in \mathbf{P}$ *is a predicate with* $\mathsf{ar}(r) = n$.
– *An* existential rule *(or simply* rule *in the context of this paper) is a formula of the form*

$$\forall \mathbf{x}.(B_1 \wedge \ldots \wedge B_k \rightarrow \exists \mathbf{y}.H_1 \wedge \ldots \wedge H_l),$$

*where* $B_1, \ldots, B_k, H_1, \ldots, H_l$ *are atoms all of whose variables are in the scope of some quantifier, and where no variable occurs more than once in* $\mathbf{x}, \mathbf{y}$.[1] *We use sets of atoms as a convenient notation for conjunctions of atoms. A* Datalog rule *is a rule with no existential quantifiers. A rule with* $k = 0$ *is called a* fact *(a head that is unconditionally true), and a rule with* $l = 0$ *is called a* constraint *(a body that must never be true).*

*The premise of a rule is called the* body *while the conclusion is called the* head*. Since all variables in rules are quantified, we will often omit the explicit preceding universal quantifier.*

*A rule set* $\Sigma$ *is* renamed appart *if each variable name is bound in at most one quantifier in* $\Sigma$.

The rule language hereby introduced is a syntactic fragment of first-order predicate logic, and we consider it under the according semantics. This also means that every rule set is semantically equivalent to one that is renamed appart. Moreover, note that we do not exclude *non-safe rules*, i.e. rules with universally quantified variables that occur in the head but not in the body; all of our results apply in any case.

**Definition 2.** *Let* $\Sigma$ *be a set of rules. We call* $\Sigma$ satisfiable *if it has a model according to the standard semantics of first-order logic. Two rule sets* $\Sigma$ *and* $\Sigma'$ *are* equisatisfiable *if either both or none of them is satisfiable. A* boolean conjunctive query *(BCQ) is a set of atoms. We say that a BCQ* $Q$ *is entailed by* $\Sigma$*, if* $\exists \mathbf{x}.Q$ *(with* $\mathbf{x}$ *containing all variables occurring in* $Q$*) is a logical consequence of* $\Sigma$ *according to standard first-order logic semantics.*

Checking satisfiability and BCQ entailment for unrestricted existential rules is undecidable [12, 5] even with very strong restrictions on the vocabulary or the number of rules [2]. Therefore, a large body of work has been devoted to the identification of restricted rule languages which retain decidability and still allow for sufficient expressiveness. A generic tool for establishing decidability results is the *chase* introduced by [17] and extended to query containment by [16] . Intuitively the chase procedure starts with a given set of factual data (ground facts) and "applies" rules in a production rule style by introducing new domain elements whenever required by an existentially quantified variable in a rule head. In general, termination of this procedure cannot be guaranteed, and an infinite set of new domain elements and facts may be created.

Many of the decidable rule classes come about by establishing properties about the chase they create. Finiteness of the chase is a straightforward criterion for ensuring decidability, and rule sets with this property are called *finite extension sets* [2]. This

[1] We freely use $\mathbf{x}$, $\mathbf{t}$, etc. to denote vectors of the form $\langle x_1, \ldots, x_n \rangle$, $\langle t_1, \ldots, t_n \rangle$, etc. throughout this paper.

criterion is undecidable in general, but several sufficient conditions on rule sets for chase-finiteness have been identified. Pure Datalog (also known as *full implicational dependencies* [12] or *total TGDs* [5]) is an immediate case, as no new domain elements are created at all. A more elaborate concept is (weak) acyclicity [14, 15] which we review and extend in Section 3. Another approach that pursues a similar goal by different means is to require acyclicity of the *graph of rule dependencies* introduced by [3] .

An even more relaxed condition than finiteness of the chase is that the (possibly infinite) chase enjoys a variant of the bounded treewidth property, leading to *bounded treewidth sets* [2]. Decidability of BCQ entailment follows from known decidability results for first-order logic theories with the bounded treewidth model property [13]. Again rules with this property are not recognisable in general, but a variety of sufficient conditions has been established. The most prominent examples are a number of *guardedness* conditions that we review and extend in Section 5.

Independently of the chase, other decidability criteria can be established by considering rewritings of the query in a backward-chaining manner. In analogy to the finite chase condition, one can define *finite unification sets* where this rewriting procedure terminates and yields a finite set of rewritten queries [2]. First-order rewritability also implies a sub-polynomial $AC_0$ data complexity for BCQ entailment checking. Again, recognising finite unification sets is undecidable, and various decidable sublanguages are known. Examples include *atomic-hypothesis rules* and *domain restricted rules* [2], *linear Datalog+/–* [7], *sticky sets of TGDs*, and *sticky-join sets of TGDs* [8, 9].

## 3 Joint Acyclicity

This section introduces *joint acyclicity*, which is a proper generalisation of the following notion of *weak acyclicity* [14, 15]:

**Definition 3.** *For a set of rules $\Sigma$, the* dependency graph *is a directed graph that has the positions of predicates in $\Sigma$ as its nodes. For every rule $\rho \in \Sigma$, and every variable $x$ at position $\langle r, p\rangle$ in the head of $\rho$, the graph contains edges as follows:*

- *If $x$ is universally quantified, and $x$ occurs in a body atom at position $\langle s, q\rangle$, there is an edge from $\langle s, q\rangle$ to $\langle r, p\rangle$.*
- *If $x$ is existentially quantified, and the body of $\rho$ contains a (necessarily universally quantified) variable $y$ at $\langle r, p\rangle$, then there is a* special edge *from $\langle s, q\rangle$ to $\langle r, p\rangle$.*

*$\Sigma$ is* weakly acyclic *if its dependency graph has no cycle going through a special edge.*

Intuitively, non-special edges encode the possible passing of values in bottom-up reasoning, whereas special edges encode the dependency between the premise that a rule was applied to and the new individuals that the application of this rule entails. A cycle over special edges may indicate that newly invented values can recursively be used in premises which require the invention of further values *ad infinitum*. For instance, the rule

$$r(x, y) \rightarrow \exists z.r(y, z) \tag{1}$$

may lead to the construction of an infinite $r$-chain of new elements, and indeed the dependency graph has a special edge from $\langle r, 2\rangle$ to itself. But weak acyclicity also excludes cases where no infinite recursion would occur:

$$r(x, y) \wedge c(y) \rightarrow \exists z.r(y, z) \tag{2}$$

The dependency graph contains the same cycle as before, yet the rule cannot be applied recursively since invented values are not required to belong to $c$. Note that this remains true even if there are other rules with existentially quantified variables at $\langle c, 1\rangle$. We capture this by shifting our focus from positions to variables (which can occur in multiple positions):

**Definition 4.** *Consider a renamed appart set of rules $\Sigma$. For a variable $x$, let $\Pi^B_x$ ($\Pi^H_x$) be the set of all positions where $x$ occurs in the body (head) of a – necessarily unique – rule. Now for any existentially quantified variable, let $\Omega_x$ be the smallest set of positions such that (1) $\Pi^H_x \subseteq \Omega_x$, and (2) $\Pi^H_y \subseteq \Omega_x$ for every universally quantified variable $y$ with $\Pi^B_y \subseteq \Omega_x$.*

*The* existential dependency graph *of $\Sigma$ has the existentially quantified variables of $\Sigma$ as its nodes. There is an edge from $x$ to $y$ if the rule where $y$ occurs contains a universally quantified (body) variable $z$ with $\Pi^B_z \subseteq \Omega_x$.*

*$\Sigma$ is* jointly acyclic *if its existential dependency graph is acyclic.*

Thus $\Omega_x$ contains the positions in which values invented for $x$ may appear. This captures the effect of non-special edges in Definition 3, whereas special edges correspond to edges in the existential dependency graph. Definition 3 is obtained by modifying condition (2) in Definition 4 to require $\Pi^B_y \cap \Omega_x \neq \emptyset$ instead of $\Pi^B_y \subseteq \Omega_x$. This states that a value is propagated by a rule if it satisfies *some* – instead of *all* – of the rule's premises. Joint acyclicity therefore appears to be the more natural condition.

The following rule is jointly acyclic (as a singleton set) but not weakly acyclic: its existential dependency graph does not have any edges whereas its dependency graph is a clique of special edges.

$$r(x, y) \wedge s(x, y) \rightarrow \exists v,w.r(x,v) \wedge r(w, y) \wedge s(x,w) \wedge s(v, y) \tag{3}$$

In spite of this generalisation, joint acyclicity is easy to recognise. Detecting cycles in a directed graph and checking inclusion of a position in $\Omega_x$ is possible in polynomial time. The latter problem is also hard for P since propositional Horn logic entailment can be expressed using unary predicates with a single variable to encode propositions.

## 4 Reducing Jointly Acyclic Variables

We now present a method for eliminating existential quantifiers from rule sets. Applied iteratively to jointly acyclic rules, this procedure yields a Datalog program that faithfully represents all consequences of the original rule set. This establishes decidability and optimal complexity bounds for jointly acyclic rules. For the general case, the procedure still allows semantically faithful simplifications of rules that can be used to extend other decidable rule languages as in Section 6.

Our transformation simulates Skolemisation, the replacement of existentially quantified variables with Skolem terms, where we "flatten" function terms to represent them in Datalog. For example, Skolemising the rule $r(x, y) \rightarrow \exists v.s(x, v)$ yields $r(x, y) \rightarrow s(x, f(x, y))$ where $f$ is a fresh function symbol. We express this without functions by considering $f$ as a constant and replacing $s$ by a predicate $s'$ of higher arity: $r(x, y) \rightarrow s'(x, f, x, y)$. Other predicates may need to be extended analogously in positions where the Skolem term might be relevant; those are exactly the positions in $\Omega_v$. Conversely, some uses of $s$ may not require all the new positions, and we use a special symbol $\square$ as a filler. For example, a fact $s(a, b)$ is represented as $s'(a, b, \square, \square)$.

**Definition 5.** *Consider a renamed appart rule set $\Sigma$, such that there is an existentially quantified variable $x$ that does not have incoming edges in the existential dependency graph.*

*Let $k$ be the number of universally quantified variables in the rule containing $x$. For a predicate $r$ define $n_r := \#\{\langle r, p\rangle \in \Omega_x \mid 1 \leq p \leq \mathsf{ar}(r)\}$ where $\#$ denotes set cardinality. If $n_r > 0$ let $\hat{r}$ denote a fresh predicate of arity $\mathsf{ar}(\hat{r}) = \mathsf{ar}(r) + n_r k$; if $n_r = 0$ let $\hat{r}$ denote $r$. Let $f$ and $\square$ be fresh constant symbols.*

*$\Sigma_x$ is the set of rules that contains, for each rule $\rho \in \Sigma$, the rule $\rho_x$ that is obtained by replacing each atom $r(t_1, \ldots, t_{\mathsf{ar}(r)})$ in $\rho$ by the atom $\hat{r}(\mathbf{s_1}, \ldots, \mathbf{s}_{\mathsf{ar}(r)})$ where the term vectors $\mathbf{s_i}$ are defined as follows:*

- *If $\langle r, i\rangle \notin \Omega_x$ then $\mathbf{s_i} := t_i$.*
  *For the remaining cases, assume that $\langle r, i\rangle \in \Omega_x$.*
- *If $t_i = x$ then $\mathbf{s_i} := \langle f, y_1, \ldots, y_k\rangle$ where $y_1, \ldots, y_k$ are all universally quantified variables in the rule.*
- *If $t_i = y$ is universally quantified and occurs only in positions in $\Omega_x$, then $\mathbf{s_i} := \langle y_0, y_1, \ldots, y_k\rangle$ where the same fresh universally quantified variable names $y_j$ are used in all replacements of $y$ but nowhere else.*
- *In all other cases, $\mathbf{s_i} := \langle t_i, \square, \ldots, \square\rangle$ where this is a vector of length $k + 1$.*

*Quantifiers for $\rho$ are updated accordingly: new universal quantifiers are introduced for all variables of the form $y_j$, and the existential quantifier for $x$ is deleted.*

*Given a boolean conjunctive query $Q$ over the signature of $\Sigma$, the BCQ $Q_x$ is defined as the body of the rule $\forall \mathbf{y}.Q_x \rightarrow$ that is obtained by applying the above transformation to the rule $\forall \mathbf{y}.Q \rightarrow$.*

Note that this definition is well. In particular, for each $r$ we find that $n_r$ of the vectors $\mathbf{s_i}$ are of length $k + 1$, and all others are of length 1, yielding the required $\mathsf{ar}(r) + n_r k$ arguments of $\hat{r}$. Applying this transformation to $v$ in rule (3), we have $k = 2$ and $\Omega_v = \{\langle r, 2\rangle, \langle s, 1\rangle\}$, and so obtain:

$$\hat{r}(x, y, \square, \square) \wedge \hat{s}(x, \square, \square, y) \rightarrow \exists w.\hat{r}(x, f, x, y) \wedge \hat{r}(w, y, \square, \square) \wedge \hat{s}(x, \square, \square, w) \wedge \hat{s}(f, x, y, y) \quad (4)$$

Now the main correctness result for this transformation is:

**Theorem 1.** *Given a set of rules $\Sigma$ and a variable $x$ as in Definition 5, $\Sigma$ is satisfiable if and only if $\Sigma_x$ is satisfiable. Moreover, a BCQ $Q$ over the signature of $\Sigma$ is entailed by $\Sigma$ if and only if $Q_x$ is entailed by $\Sigma_x$.*

*Proof.* The claim for BCQs is reduced to the claim of equisatisfiability by noting that $\Sigma \cup \{\forall \mathbf{y}.Q \rightarrow\}$ is satisfiable iff $\Sigma$ entails $Q$. It remains to show equisatisfiability.

For the one direction, assume that $\mathcal{I}$ is a model of $\Sigma$ with domain $\Delta^{\mathcal{I}}$, and construct a model $\mathcal{J}$ of $\Sigma_x$ as follows. We use the notation as in Definition 5. The domain $\Delta^{\mathcal{J}}$ of $\mathcal{J}$ is defined as $\Delta^{\mathcal{I}} \cup \{\square, f\}$ where $\square$ and $f$ are assumed to be distinct from any element in $\Delta^{\mathcal{I}}$. For a predicate $r$, any tuple of $\mathsf{ar}(\hat{r})$ elements from $\Delta^{\mathcal{J}}$ can be written as $\langle \mathbf{e_1}, \dots, \mathbf{e}_{\mathsf{ar}(r)} \rangle$ with vectors $\mathbf{e_1}$ of length 1 or $k+1$.

Assume that the rule that contains $x$ is of the form $\forall \mathbf{y}.\varphi \rightarrow \exists \mathbf{z}.\psi$. We define a mapping $\iota$ from such vectors to $\Delta^{\mathcal{I}}$:

- If $\mathbf{e_i} = \langle \epsilon \rangle$, $\epsilon \in \Delta^{\mathcal{I}}$ then $\iota(\mathbf{e_i}) \coloneqq \epsilon$.
- If $\mathbf{e_i} = \langle f, \epsilon_1, \dots, \epsilon_k \rangle$ with $\epsilon_i \in \Delta^{\mathcal{I}}$, then let $\mathcal{Z}$ be a variable assignment for $\mathcal{I}$ that maps $y_i$ (the universally quantified variables in $\forall \mathbf{y}.\varphi \rightarrow \exists \mathbf{z}.\psi$) to $\epsilon_i$.
  - If $\mathcal{I}, \mathcal{Z} \not\models \varphi$, let $\iota(\mathbf{e_i}) \in \Delta^{\mathcal{I}}$ be arbitrary.
  - If $\mathcal{I}, \mathcal{Z} \models \varphi$ then there is a variable assignment $\mathcal{Z}'$ for $\mathcal{I}$ that agrees with $\mathcal{Z}$ on all variables $y_i$, and such that $\mathcal{I}, \mathcal{Z}' \models \psi$. Define $\iota(\mathbf{e_i}) \coloneqq \mathcal{Z}'(x)$.
- If $\mathbf{e_i} = \langle \epsilon, \square, \dots, \square \rangle$ (length $k+1$) then $\iota(\mathbf{e_i}) \coloneqq \epsilon$.
- In all other cases, let $\iota(\mathbf{e_i}) \in \Delta^{\mathcal{I}}$ be arbitrary.

Moreover, we define the $\mathcal{J}$ extension of a predicate $\hat{r}$ by setting $\langle \mathbf{e_1}, \dots, \mathbf{e}_{\mathsf{ar}(r)} \rangle \in \hat{r}^{\mathcal{J}}$ iff $\langle \iota(\mathbf{e_1}), \dots, \iota(\mathbf{e}_{\mathsf{ar}(r)}) \rangle \in r^{\mathcal{I}}$. Finally, define $\square^{\mathcal{J}} \coloneqq \iota(\langle \square \rangle)$ and $f^{\mathcal{J}} \coloneqq \iota(\langle f \rangle)$ (those values have been fixed arbitrarily when defining $\iota$).

Then $\mathcal{J}$ is a model of $\Sigma_x$. This is easily checked for an arbitrary rule in $\Sigma_x$: if $\mathcal{J}$ satisfies the body under some variable assignment $\mathcal{Z}$, then $\iota$ can be used to construct a variable assignment under which $\mathcal{I}$ satisfies the original rule body in $\Sigma$. One concludes that the rule head in $\Sigma$ is also satisfied and can use $\iota$ to find suitable tuples to establish this in $\Sigma_x$. This is immediate in most cases. Head atoms that involve $x$ are special since the corresponding head in $\Sigma_x$ contains the constant $f$ in this case. The construction of $\iota$ above ensures that $\iota$ is based on an assignment $\mathcal{Z}'$ for $\mathcal{I}$ that satisfies the rule, and this very assignment can also be used for finding values for any other existentially quantified variables that occur in this rule head of $\Sigma_x$.

For the other direction, assume that $\mathcal{J}$ is a model of $\Sigma_x$. We construct a model $\mathcal{I}$ of $\Sigma$ as follows. The elements of $\Delta^{\mathcal{I}}$ are all tuples of elements from $\Delta^{\mathcal{J}}$ that contain either 1 or $k+1$ elements. For a predicate $r$ of $\Sigma$ we set $\langle \mathbf{e_1}, \dots, \mathbf{e}_{\mathsf{ar}(r)} \rangle \in r^{\mathcal{I}}$ iff $\langle \mathbf{e_1}, \dots, \mathbf{e}_{\mathsf{ar}(r)} \rangle \in \hat{r}^{\mathcal{J}}$. It is easy to see that $\mathcal{I} \models \Sigma$. □

We can thus apply Definition 5 iteratively, where Theorem 1 ensures that correctness is preserved. To this end, it is important that the iterative reduction also preserves joint acyclicity:

**Theorem 2.** *Consider a rule set $\Sigma$, and a variable $x$ as in Definition 5. The variables $y \neq x$ without incoming edges in the existential dependency graph of $\Sigma$ do not have incoming edges in the existential dependency graph of $\Sigma_x$ either. Moreover, $\Sigma$ is jointly acyclic if and only if $\Sigma_x$ is jointly acyclic.*

*Proof.* The set of existentially quantified variables decreases monotonically during the reduction: newly introduced variables are not existentially quantified. Both claims are

established by showing that the reduction does not lead to new edges in the existential dependency graph. Given an existentially quantified variable $y \neq x$, let $\Omega_y$ and $\Omega'_y$ denote the respective sets of Definition 4 before and after the reduction, respectively. None of the universal variables in the rule that contains $x$ occurs in any of the sets $\Omega_y$ since $x$ does not have any incoming edges in the existential dependency graph. The reduction step introduces new predicates that may have $(k + 1)$ positions where the original predicates had a single position. However, existentially quantified variables $y \neq x$ only ever occur at the first of these positions, so this position is the starting point for computing $\Omega'_y$. Considering the replacements of terms by term vectors in Definition 5, it is clear that $\Omega'_y$ can be obtained from $\Omega_y$ by simply mapping original positions to the first of the new positions. Thus each edge in the existential dependency graph after the reduction step corresponds to such an edge before the reduction. In particular, no new cycles are introduced. □

The previous theorem ensures that the set of variables that can be eliminated by applying Definition 5 iteratively is not affected by the order in which variables are reduced in case there is more than one variable without incoming edges. Yet, iterative reductions may yield syntactically different results depending on the order of application. This non-determinism is inessential for our considerations, so we use $\mathsf{ja}(\Sigma)$ to denote an arbitrary but fixed rule set obtained by iteratively applying Definition 5 until it is no longer applicable.

**Theorem 3.** *If $\Sigma$ is a jointly acyclic, renamed appart set of rules $\Sigma$ then* $\mathsf{ja}(\Sigma)$ *is a Datalog program.*

*Proof.* The claim can be shown inductively. If a rule set is jointly acyclic then its existential dependency graph must clearly contain a variable $x$ without incoming edges, so Definition 5 is initially applicable. The induction step is established by Theorem 2. □

Before stating the main complexity result of this section, we provide a more precise estimate of the increase in size that is caused by the transformation. Importantly, the exponential blow-up is caused by chains of dependencies in the existential dependency graph, not by the size of the rule set in general.

**Theorem 4.** *Given a renamed appart rule set $\Sigma$, the set* $\mathsf{ja}(\Sigma)$ *contains the same number of rules as $\Sigma$, and the same number of head and body atoms in each rule. The number of variables per rule in* $\mathsf{ja}(\Sigma)$ *is bounded by a function that is exponential in the maximum directed path length in the existential dependency graph of $\Sigma$, and polynomial in the size of $\Sigma$.*

*Proof.* It is immediate that the reduction of Definition 5 changes neither the number of rules nor the number of atoms per rule body or head. The only new variables introduced in the reduction step are the $(k + 1)$ fresh variables used to replace variables $y$ that occurred in positions in $\Omega_x$ only (where $x$ is the existentially quantified variable that is reduced). We note the following properties:

(1) At most one of these fresh variables can be on a position that is in $\Omega_z$ of another existential variable after the translation.

(2) If the $(k+1)$ fresh variables are introduced in the body of a rule with an existentially quantified variable $z$ (that could possibly be reduced later on), then the existential dependency graph contains an edge from $x$ to $z$.

From (1) we conclude that a rule can obtain at most $n \times k$ additional universally quantified variables in the translation step, where $n$ is the maximal number of such variables per rule in $\Sigma$. Importantly, $n$ is constant throughout the reduction due to (1), whereas $k$ depends on the current step. So the number of variables per rule after a series of reduction steps with $k$ values $k_1, \ldots, k_m$ is bounded by $n + k_1 n + \ldots + k_m n = n(1 + k_1 + \ldots + k_m)$. The number $m$ of reductions is bounded by the overall number of existentially quantified variables in $\Sigma$.

From (2) we conclude that the maximal number of universally quantified variables in a rule that contains an existential quantifier can only increase if there is a corresponding edge in the existential dependency graph. In this case it increases by at most $n \times k$ (leading to a new maximum of $(n+1)k$), where $k$ is initially bounded by $n$. Hence, values $k_j$ are bounded by $(n+1)^d$ where $d$ is the maximum directed path length in the existential dependency graph of $\Sigma$. This is the maximal number of universally quantified variables in any rule with an existential quantifier at any stage of the reduction. For general rules, our earlier bound can be simplified by noting that $k_j \leq (n+1)^d$ to obtain $n(1 + m(n+1)^d) \leq (1+m)(n+1)^{(d+1)}$. Since $m$ and $n$ are bounded by the size of $\Sigma$, this establishes the claim. □

**Theorem 5.** *Deciding whether BCQ $Q$ is entailed by a jointly acyclic set of rules $\Sigma$ is* 2ExpTime*-complete for combined complexity,* ExpTime*-complete if the maximal length of a path in the existential dependency graph is bounded, and* P*-complete in data complexity.*

*Proof.* The set $\Sigma$ can be transformed into a renamed appart set of rules in linear time. Inclusion then follows by Theorem 3 and 4, together with the well-known complexities of BCQ answering for Datalog. Hardness for 2ExpTime follows from the respective hardness result for weakly acyclic rules [9]. Hardness for ExpTime and P follows again from the respective hardness of Datalog. □

## 5 Jointly Frontier-Guarded Rules

A large class of existential rules for which query answering is decidable are based on the idea of *guardedness* [1], the requirement that all or some of the universally quantified variables of a rule appear together in a single "guard" atom. Requiring guards only for variables that also appear in the head (the "frontier") yields *frontier-guarded rules* [2]. Both notions can be generalised by not requiring guards for variables that cannot possibly represent existentially introduced elements. This idea has been used to arrive at *weakly guarded rules* [6] and *weakly frontier-guarded rules* [2]. In this section, we generalise the latter to fit more naturally to our definitions in Section 3, and we establish basic complexity results.

**Definition 6.** *Consider a set of rules $\Sigma$. A position $\langle r, i \rangle$ is* affected *if (1) $\Sigma$ contains an existentially quantified variable on position $\langle r, i \rangle$, or (2) $\Sigma$ contains a universally*

*quantified variable x on position $\langle r, i\rangle$ in the head of a rule where x occurs on an affected position in its body. A position $\langle r, i\rangle$ is* jointly affected *if $\langle r, i\rangle \in \Omega_x$ for some variable x in $\Sigma$ (see Definition 4).*

*A variable x in a rule $\rho = \forall \mathbf{x}.\varphi \to \exists \mathbf{y}.\psi \in \Sigma$ is* universal *if it occurs in* $\mathbf{x}$*,* affected *if it occurs on some affected position in $\varphi$,* jointly affected *if it occurs only on jointly affected positions in $\varphi$,* frontier *if it occurs $\varphi$ and in $\psi$. The sets of all such variables are denoted $X^u_\rho$, $X^a_\rho$, $X^{ja}_\rho$, $X^f_\rho$.*

*The rule $\rho$ is X-guarded for a set X of variables, if all $x \in X$ occur together in one atom in $\varphi$. Relevant notions are:* guarded *($X = X^u_\rho$),* frontier-guarded *($X = X^f_\rho$),* weakly guarded *($X = X^a_\rho$),* weakly frontier-guarded *($X = X^a_\rho \cap X^f_\rho$),* jointly guarded *($X = X^{ja}_\rho$),* jointly frontier-guarded *($X = X^{ja}_\rho \cap X^f_\rho$). The set $\Sigma$ is X-guarded if all rules $\rho \in \Sigma$ are.*

The relation of these notions follows from the observation that $X^u_\rho \supseteq X^f_\rho$ and $X^u_\rho \supseteq X^a_\rho \supseteq X^{ja}_\rho$, e.g. every weakly guarded rule is also jointly frontier-guarded. The combined complexity of BCQ answering for guarded and weakly guarded rules is known to be 2ExpTime-complete [6]. Hardness carries over to the frontier-guarded cases, but upper complexity bounds for these languages have been open until very recently. We cite the following result from a concurrent anonymous submission to this conference. We reproduce the respective proof here without claiming this as our own contribution.

**Proposition 1.** *Deciding whether a BCQ Q is entailed by a frontier-guarded set of rules $\Sigma$ is* 2ExpTime*-complete for combined complexity.*

*Proof.* Hardness follows from the known hardness result for guarded rules [6].

To show inclusion, we make use of the result of [4] that deciding entailment of unions of BCQs in the guarded fragment (GF) of first-order logic is 2ExpTime-complete. This result can be used to prove 2ExpTime inclusion for frontier-guarded BCQ entailment.

Every frontier-guarded rule $\rho$ with a non-empty body $\mathsf{body}(\rho)$ can be translated into two rules, where one is guarded and one is Datalog: given an (arbitrary but fixed) frontier guard $p(\mathbf{t}) \in \mathsf{body}(\rho)$, we introduce a new $n$-ary predicate $p_\rho$ and let $\mathsf{sep}(\rho)$ be the set of the two rules $\check{\rho} := \mathsf{body}(\rho) \to p_\rho(\mathbf{t})$ and $\hat{\rho} := p_\rho(\mathbf{t}) \to \mathsf{head}(\rho)$. For rules $\rho$ with empty body, i.e. facts, we set $\mathsf{sep}(\rho) := \{\rho\}$. It is immediate that for any frontier-guarded rule set $\Sigma$, we have $\Sigma \models Q$ exactly if $\bigcup_{\rho\in\Sigma} \mathsf{sep}(\rho) \models Q$.

Obviously, $\hat{\rho}$ is guarded (and hence also lies in GF) while $\check{\rho}$ is Datalog and frontier-guarded (but not necessarily guarded). However, we can transform $\check{\rho}$ as follows, where we use $\mathbf{x}$ and $\mathbf{y}$ to denote the variables in $\mathsf{body}(\rho)$ and in $\mathbf{t}$, respectively, and introduce a fresh predicate $p'_\rho$:

$$
\begin{aligned}
& \forall \mathbf{x}.(\mathsf{body}(\rho) \to p_\rho(\mathbf{t})) \\
\text{iff}\ & \neg\exists \mathbf{x}.(\mathsf{body}(\rho) \wedge \neg p_\rho(\mathbf{t})) \\
\text{iff}\ & (\neg\exists \mathbf{x}.(\mathsf{body}(\rho) \wedge p'_\rho(\mathbf{t}))) \wedge (\forall \mathbf{y}.(p(\mathbf{t}) \wedge \neg p_\rho(\mathbf{t}) \to p'_\rho(\mathbf{t}))) \\
\text{iff}\ & \underbrace{(\neg\exists \mathbf{x}.(\mathsf{body}(\rho) \wedge p'_\rho(\mathbf{t})))}_{=:\check{\rho}_1} \wedge \underbrace{(\forall \mathbf{y}.(p(\mathbf{t}) \to p'_\rho(\mathbf{t}) \vee p_\rho(\mathbf{t})))}_{=:\check{\rho}_2}
\end{aligned}
$$

Hence $\Sigma \models Q$ iff

$$\{\hat{\rho}, \check{\rho}_2 \mid \rho \in \Sigma\} \cup \{\check{\rho}_1 \mid \rho \in \Sigma\} \models Q \qquad (\dagger)$$

where the first set is in GF and the second consists of negated existentially quantified conjunctions of atoms. Hence we can conceive every $\check{\rho}_1$ as a negated conjunctive query $\neg Q_\rho$. Consequently we have

$$\{\check{\rho}_1 \mid \rho \in \Sigma\} \equiv \{\neg Q_\rho \mid \rho \in \Sigma\} \equiv \bigwedge_{\rho \in \Sigma} \neg Q_\rho \equiv \neg \bigvee_{\rho \in \Sigma} Q_\rho$$

which allows to rephrase (†) as

$$\{\hat{\rho}, \check{\rho}_2 \mid \rho \in \Sigma\} \models Q \vee \bigvee_{\rho \in \Sigma} Q_\rho$$

leaving us with a GF theory on the left-hand side and a union of boolean conjunctive queries on the right-hand side. As the translation is clearly linear, we have thus shown 2ExpTime-inclusion for BCQ entailment for frontier-guarded rules. □

The anonymous work where this result has first been established also contains a proof showing BCQ answering for weakly frontier-guarded rules to be in 2ExpTime. We extend this result to our new notion of jointly guarded and jointly frontier-guarded rules. Our respective proofs are new and original, though based on similar ideas. Namely, we observe that variables that are not jointly affected may never represent elements that are introduced existentially. Thus, their assignments correspond to constant symbols that could be substituted instead. A naïve use of this idea yields exponentially many *partially grounded* rules with constants used in all possible combinations.

A polynomial reduction is possible by extending the arguments of all predicates to contain parameters for all variables that are not jointly affected. These parameters then guard all such variables in rules. Bindings for the added parameters can only be inferred by auxiliary rules that allow arbitrary constants to be substituted for variables. These ideas are combined to the following definition.

**Definition 7.** *For a renamed appart rule set $\Sigma$, let $\mathbf{z} = \langle z_1, \dots, z_n \rangle$ be a list of all variables in $\Sigma$ that are not jointly affected, and let $\tilde{r}$ be a fresh predicate of arity $\mathsf{ar}(r) + n$ for each predicate $r$ of $\Sigma$. The rule set $\mathsf{guard}(\Sigma)$ consists of:*

*(1) for each rule $\rho \in \Sigma$ with non-empty body, a rule $\rho' \in \mathsf{guard}(\Sigma)$ obtained by replacing every atom $r(t_1, \dots, t_{\mathsf{ar}(r)})$ (with terms $t_i$) by the atom $\tilde{r}(t_1, \dots, t_{\mathsf{ar}(r)}, z_1, \dots, z_n)$, where all variables $z_i$ are universally quantified,*

*(2) for each rule $\rho \in \Sigma$ with an empty body (i.e. generalised fact), a rule $\rho' \in \mathsf{guard}(\Sigma)$ that is obtained by replacing every atom $r(t_1, \dots, t_{\mathsf{ar}(r)})$ (with terms $t_i$) by the atom $\tilde{r}(t_1, \dots, t_{\mathsf{ar}(r)}, c, \dots, c)$ where $c$ is an arbitrary constant,*

*(3) for each predicate $r$ of $\Sigma$, each $i \in \{1, \dots, n\}$, and each constant symbol $c$, a rule*

$$\tilde{r}(x_1, \dots, x_{\mathsf{ar}(r)}, z_1, \dots, z_i, \dots, z_n) \to \tilde{r}(x_1, \dots, x_{\mathsf{ar}(r)}, z_1, \dots, c, \dots, z_n),$$

*(4) for each predicate $r$ of $\Sigma$, a rule*

$$\tilde{r}(x_1, \dots, x_{\mathsf{ar}(r)}, z_1, \dots, z_n) \to r(x_1, \dots, x_{\mathsf{ar}(r)}),$$

*where all variable names $x_i$ are fresh.*

**Theorem 6.** *A BCQ $Q$ is entailed by a renamed appart rule set $\Sigma$ iff $Q$ is entailed by* $\mathsf{guard}(\Sigma)$.

*Proof.* A model $\mathcal{I}$ is *strictly larger* than a model $\mathcal{J}$ if (a) both have the same domain $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}}$, (b) $c^{\mathcal{J}} = c^{\mathcal{I}}$ for all constants $c$, (c) $r^{\mathcal{J}} \subseteq r^{\mathcal{I}}$ for all predicates $r$, and (d) $s^{\mathcal{J}} \subsetneq s^{\mathcal{I}}$ for at least one predicate. It is well known that entailment is the same when restricting to models that are minimal w.r.t. this order, i.e. to models $\mathcal{I}$ that are not strictly larger than any other model.

Now the claim follows from the following correspondence of models: (1) every minimal model of $\mathsf{guard}(\Sigma)$ is a model of $\Sigma$ (when restricted to the predicates in $\Sigma$), and (2) every model of $\Sigma$ can be extended to a model of $\mathsf{guard}(\Sigma)$ by defining a suitable interpretation for the new predicates.

For (1), let $\mathcal{I}$ be a minimal model of $\mathsf{guard}(\Sigma)$ and consider a rule $\forall \mathbf{x}.\psi \to \exists \mathbf{y}.\varphi \in \Sigma$. First consider the case that $\psi$ is non-empty, and let $\mathcal{Z}$ be a variable assignment for $\mathcal{I}$ such that $\mathcal{I}, \mathcal{Z} \models \psi$. Let $r(\mathbf{x})$ be an atom in $\psi$. We want to show that there is a variable assignment $\mathcal{Z}'$ that agrees with $\mathcal{Z}$ on all variables of $\mathbf{x}$, and such that $\mathcal{I}, \mathcal{Z}' \models \psi'$ where $\forall \mathbf{xz}.\psi' \to \exists \mathbf{y}.\varphi'$ is the rule in $\mathsf{guard}(\Sigma)$ that was generated by (1) of Definition 7. To this end, first note that by minimality and rules (4) of Definition 7, $\mathcal{I}, \mathcal{Z} \models r(\mathbf{x})$ implies that there is some $\langle e_1, \dots, e_{\mathsf{ar}(r)}, f_1, \dots, f_n \rangle \in \tilde{r}^{\mathcal{I}}$ with $\langle e_1, \dots, e_{\mathsf{ar}(r)} \rangle = \mathcal{Z}(\mathbf{x})$.

Now we need to show that this can be used to find an assignment $\mathcal{Z}'$ such that $\mathcal{I}, \mathcal{Z}' \models \tilde{r}(\mathbf{x}, \mathbf{z})$. It helps to note that the existence of $\langle e_1, \dots, e_{\mathsf{ar}(r)}, f_1, \dots, f_n \rangle \in \tilde{r}^{\mathcal{I}}$ together with rules (3) lets us find according tuples with $f_i$ replaced by arbitrary elements of the form $c^{\mathcal{I}}$ for a constant $c$ (we call such elements $c^{\mathcal{I}}$ *named*). Unfortunately, no elements of other forms can be assumed in place of $f_i$ (one could show this using minimality and an easy induction, but we don't require this for the proof). Hence, to find the required assignment $\mathcal{I}, \mathcal{Z}' \models \tilde{r}(\mathbf{x}, \mathbf{z})$, we must ensure that, for all variables $x_i$ that occur in $\mathbf{z}$, $\mathcal{Z}(x_i)$ is a named element. Such variables $x_i$ are exactly those that occur on a position in $r$ that is not jointly affected. We show that only named elements occur on such positions inductively over the derivation steps of a bottom-up application of the rules, where we exploit that by minimality of $\mathcal{I}$, predicate extensions only contain element tuples for which there is such a bottom-up proof. The base case is given by generalised facts: they have constant symbols on all positions that are not jointly affected. For the induction step, consider any rule with non-empty body. Positions in the head that are not jointly affected must contain a constant (then the claim is immediate) or a variable that occurs on some body position that is not jointly affected (then the claim follows by induction).

The previous induction shows that we can find a variable assignment $\mathcal{Z}'$ that agrees with $\mathcal{Z}$ on $\mathbf{x}$ and such that $\mathcal{I}, \mathcal{Z}' \models \tilde{r}(\mathbf{x}, \mathbf{z})$. As the values of $\mathcal{Z}'$ on variables not in $\mathbf{x}$ can be arbitrary named elements, the same assignment $\mathcal{Z}'$ works for all atoms of $\psi$ so we find $\mathcal{I}, \mathcal{Z}' \models \psi'$ as desired. This implies $\mathcal{I}, \mathcal{Z}'' \models \varphi'$ for some assignment $\mathcal{Z}''$ that agrees with $\mathcal{Z}'$ on all variables other than possibly $\mathbf{y}$. From the construction of $\varphi$ and rules (4), we conclude $\mathcal{I}, \mathcal{Z}'' \models \varphi$ as claimed, where we note that the variables $x_i$ and $z_j$ in the rules (4) are distinct. The same conclusion follows directly for the remaining case that $\psi$ is empty. Thus, $\mathcal{I}$ is a model of $\Sigma$.

For (2), let $\mathcal{I}$ be a model of $\Sigma$. We extend $\mathcal{I}$ to predicates of the form $\tilde{r}$ by setting $\langle e_1, \dots, e_{\mathsf{ar}(r)}, f_1, \dots, f_n \rangle \in \tilde{r}^{\mathcal{I}}$ iff $\langle e_1, \dots, e_{\mathsf{ar}(r)} \rangle \in \tilde{r}^{\mathcal{I}}$ and, for each $i \in \{1, \dots, n\}$, $f_i$ is of

the form $c^{\mathcal{I}}$ for some constant $c$. Rules of types (3) and (4) are clearly satisfied by this interpretation. Now consider a rule $\forall \mathbf{xz}.\psi' \rightarrow \exists \mathbf{y}.\varphi'$ of type (1) that was created from a rule $\forall \mathbf{x}.\psi \rightarrow \exists \mathbf{y}.\varphi \in \Sigma$. Any variable assignment $\mathcal{Z}$ with $\mathcal{I}, \mathcal{Z} \models \psi'$ satisfies $\mathcal{I}, \mathcal{Z} \models \psi$. So there is some assignment $\mathcal{Z}'$ that agrees with $\mathcal{Z}$ on all variables other than $\mathbf{y}$ such that $\mathcal{I}, \mathcal{Z}' \models \varphi$. Since $\mathcal{Z}$ and $\mathcal{Z}'$ agree on $\mathbf{z}$ as well, this implies $\mathcal{I}, \mathcal{Z}' \models \varphi'$ as required. A similar conclusion settles the case for type (3) rules, so $\mathcal{I}$ is a model of $\mathsf{guard}(\Sigma)$. □

**Theorem 7.** *Deciding whether BCQ $Q$ is entailed by a jointly guarded or jointly frontier-guarded set of rules $\Sigma$ is* 2ExpTime*-complete for combined complexity.*

*Proof.* Hardness follows from Proposition 1 for both guarded and frontier-guarded rules. For inclusion, we only need to consider jointly frontier-guarded rules. Observe that $\mathsf{guard}(\Sigma)$ is polynomial in the size of $\Sigma$, and that $\mathsf{guard}(\Sigma)$ is (frontier-)guarded whenever $\Sigma$ is jointly (frontier-)guarded. By Theorem 6, query entailment can then be decided based on the polynomially large frontier-guarded $\mathsf{guard}(\Sigma)$, which in turn is possible in 2ExpTime using Proposition 1. □

## 6 Joining Acyclicity and Guardedness

The iterative reduction in Section 4 hints at a much wider applicability of the idea of joint acyclicity, since it allows for the elimination of some existential quantifiers even in rule sets that are not jointly acyclic. This is useful if the reduced rule set belongs to a rule language for which decidability of reasoning has been established on other grounds. Here, we illustrate this idea by combining acyclicity with joint (frontier-)guardedness, and establish tight complexity bounds for related reasoning tasks.

Using the terminology of Section 5, we can say that Definition 5 eliminates jointly affected variables. To be more precise, we say that a variable in a renamed appart rule set $\Sigma$ is a *glut variable* if it occurs in a set $\Omega_x$ as in Definition 5 for a variable $x$ that is part of a cycle in the existential dependency graph. Intuitively, glut variables are those that may represent an overabundance of values, as opposed to the remaining, *non-glut variables* that can only represent finitely many values. It is easy to see that the iterative application of Definition 5 then turns non-glut variables into variables that are not jointly affected. This leads to a further generalisation of guardedness:

**Definition 8.** *A renamed appart rule set $\Sigma$ is* glut-guarded *(*glut-frontier-guarded*) if each rule of $\Sigma$ has a body atom containing all glut variables (that also occur in the head).*

The previous definition is illustrated in the following example of a glut-frontier-guarded rule set, where $c$, intuitively speaking, marks persons that are "specifically important" for us:

$$c(x) \wedge \mathit{ancestor}(x, \dot{y}) \wedge \mathit{ancestor}(\dot{y}, \dot{z}) \rightarrow \mathit{ancestor}(x, \dot{z}) \tag{5}$$

$$\mathit{parent}(\dot{x}, \dot{y}) \rightarrow \mathit{ancestor}(\dot{x}, \dot{y}) \tag{6}$$

$$c(x) \rightarrow \mathit{person}(x) \tag{7}$$

$$\mathit{person}(\dot{x}) \rightarrow \exists \dot{w}.\mathit{parent}(\dot{x}, \dot{w}) \wedge \mathit{person}(\dot{w}) \tag{8}$$

$$\mathit{sibling}(x, y) \rightarrow \exists v.\mathit{parent}(x,v) \wedge \mathit{parent}(y,v) \wedge c(v) \tag{9}$$

$$\mathit{parent}(\dot{x}, y) \wedge \mathit{sibling}(y, z) \rightarrow \mathit{uncle}(\dot{x}, z) \tag{10}$$

Information about $c$, *parent*, and *sibling* would be given in facts, while the remaining predicates are derived only. The existential dependency graph has two edges $v \to w$ and $w \to w$, where the latter cycle follows from (8). Glut variables thus are those occurring only on positions of $\Omega_w$; they are marked by a dot in the example. It is easy to verify that the example is glut-frontier-guarded. Note how $c$ is used to make $x$ in rule (5) non-glut, thus allowing a form of transitivity – a typical counter-example for all common types of guardedness. Furthermore, transitivity is not first-order rewritable, thus excluding the example from all types of finite unification sets reviewed in Section 2. Rule (10) is another illustration of the increased expressive power, since it is neither jointly frontier-guarded nor glut-guarded. Indeed, since all positions other than those of *sibling* are in $\Omega_v$, almost all variables in the example are jointly affected.

**Theorem 8.** *Deciding whether BCQ $Q$ is entailed by a glut-guarded or glut-frontier-guarded set of rules $\Sigma$ is* 3ExpTime*-complete for combined complexity.*

*Proof.* 3ExpTime hardness for glut-guarded rules is shown in Proposition 2 below, and hardness of glut-frontier-guarded rules follows from that. For inclusion, it suffices to consider glut-frontier-guarded rules. By Theorem 4, $\mathsf{ja}(\Sigma)$ is exponential in the size of $\Sigma$ (since the maximal directed path length in the existential dependency graph is linear in the size of $\Sigma$). Clearly, the only jointly affected variables in $\mathsf{ja}(\Sigma)$ are glut variables, and the guardedness condition for these variables is preserved during the iterative reduction of non-glut variables. Hence $\mathsf{ja}(\Sigma)$ is jointly frontier-guarded, and the result follows from Theorem 7. □

It remains to show the claimed hardness result. This is achieved by a reduction of the word problem for a 2ExpSpace alternating Turing machine.

**Definition 9.** *An* alternating Turing machine *(ATM) $\mathcal{M}$ is a tuple $\langle Q, \mathscr{A}, \Delta, q_0 \rangle$ where $Q = U \dot\cup E$ is the disjoint union of a finite set of* universal states *$U$ and a finite set of* existential states *$E$, $\mathscr{A}$ is a finite* alphabet *that includes a* blank *symbol $\square$, $\Delta \subseteq (Q \times \mathscr{A}) \times (Q \times \mathscr{A} \times \{l, r\})$ is a* transition relation*, and $q_0 \in Q$ is the* initial state*.*

*A (universal / existential)* configuration *of $\mathcal{M}$ is a word $\theta \in \mathscr{A}^* Q \mathscr{A}^*$ ($\mathscr{A}^* U \mathscr{A}^*$ / $\mathscr{A}^* E \mathscr{A}^*$). A configuration $\theta'$ is a* successor *of a configuration $\theta$ if one of the following holds:*

1. $\theta = w_l q \alpha \alpha_r w_r$, $\theta' = w_l \alpha' q' \alpha_r w_r$, *and* $\langle q, \alpha, q', \alpha', r \rangle \in \Delta$,
2. $\theta = w_l q \alpha$, $\theta' = w_l \alpha' q' \square$, *and* $\langle q, \alpha, q', \alpha', r \rangle \in \Delta$,
3. $\theta = w_l \alpha_l q \alpha w_r$, $\theta' = w_l q' \alpha_l \alpha' w_r$, *and* $\langle q, \alpha, q', \alpha', l \rangle \in \Delta$,

*where $q \in Q$ and $\alpha, \alpha', \alpha_l, \alpha_r \in \mathscr{A}$ as well as $w_l, w_r \in \mathscr{A}^*$. Given some natural number $s$, the possible* transitions in space *$s$ are defined by additionally requiring that $|\theta'| \leq s{+}1$.*

*The set of* accepting configurations *is the least set which satisfies the following conditions. A configuration $\theta$ is accepting iff (i) $\theta$ is a universal configuration and all its successor configurations are accepting, or (ii) $\theta$ is an existential configuration and at least one of its successor configurations is accepting. Note that universal configurations without any successors are trivially accepting.*

$\mathcal{M}$ accepts *a given word $w \in \mathscr{A}^*$ (in space $s$) iff the configuration $q_0 w$ is accepting (when restricting to transitions in space $s$).*

ATMs can solve 3ExpTime problems in doubly exponential space [11]. We thus can show 3ExpTime-hardness of BCQ entailment for glut-guarded rules by polynomially reducing the halting problem of ATMs with a doubly exponentially bounded storage space.

**Proposition 2.** *Boolean conjunctive query entailment for glut-guarded rules is* 3ExpTime*-hard even for bounded predicate arity.*

*Proof.* For any ATM $\mathcal{M} = \langle Q, \mathscr{A}, \Delta, q_0 \rangle$ and word $w \in \mathscr{A}^*$, we construct a rule set $\Sigma_{\mathcal{M},w}$ and show that acceptance of $w$ by the ATM $\mathcal{M}$ within doubly exponential space can be decided via checking fact entailment on $\Sigma_{\mathcal{M},w}$.

For the construction of $\Sigma_{\mathcal{M},w}$ we first make use of a construction introduced in [9] to provide for doubly exponentially many tape cell addresses. More precisely, for some arbitrary number $k \geq 0$, we construct a tape of length $2^{2^k}$ using a set of rules $\Sigma^{\text{tape}}_{\mathcal{M},w}$ with size proportional to $k$. An initial chain of 2 elements is constructed using constant symbols $c_0$ and $c_1$, and the following facts: $r_0(c_0)$, $r_0(c_1)$, $succ_0(c_0, c_1)$, $min_0(c_0)$, and $max_0(c_1)$. Now the following rules are organised in layers $i \in \{0, \dots, k-1\}$, where each layer combines already constructed elements to construct larger chains:

$$
\begin{aligned}
r_i(x) \wedge r_i(y) &\rightarrow \exists z.\, s_i(x, y, z) \\
s_i(x, y, z) &\rightarrow r_{i+1}(z) \\
s_i(x, y, z) \wedge s_i(x, y', z') \wedge succ_i(y, y') &\rightarrow succ_{i+1}(z, z') \\
s_i(x, y, z) \wedge s_i(x', y', z') \wedge & \\
max_i(y) \wedge min_i(y') \wedge succ_i(x, x') &\rightarrow succ_{i+1}(z, z') \\
min_i(x) \wedge s_i(x, x, y) &\rightarrow min_{i+1}(y) \\
max_i(x) \wedge s_i(x, x, y) &\rightarrow max_{i+1}(y) \\
succ_k(x, y) &\rightarrow succt(x, y) \\
succt(x, y) \wedge succt(y, z) &\rightarrow succt(x, z)
\end{aligned}
$$

The definition of $\Sigma^{\text{tape}}_{\mathcal{M},w}$ is completed by the following rules:

$$
\begin{aligned}
succ_k(x, y) &\rightarrow succt(x, y) \\
succt(x, y) \wedge succt(y, z) &\rightarrow succt(x, z)
\end{aligned}
$$

We find that this rule set projectively characterises a doubly-exponential chain: Every model $\mathcal{I}$ of $\Sigma^{\text{tape}}_{\mathcal{M},w}$ contains $2^{(2^k)}$ (not necessarily distinct) elements $\{d_1, \dots d_{2^{(2^k)}}\} \in r_k^{\mathcal{I}}$ such that $\langle d_j, d_{j+1} \rangle \in {succ_k}^{\mathcal{I}}$, and $\langle d_j, d_{j'} \rangle \in succt^{\mathcal{I}}$ for all $1 \leq j < j' \leq 2^{(2^k)}$, as well as $d_1 \in min_k^{\mathcal{I}}$ and $d_{2^{(2^k)}} \in max_k^{\mathcal{I}}$. Conversely, every set with interpretations for $succ_k$, $succt$, $min_k$, and $max_k$ with these properties can be turned into a model of $\Sigma^{\text{tape}}_{\mathcal{M},w}$ by choosing appropriate interpretations for the remaining predicates. This allows to use $\Sigma^{\text{tape}}_{\mathcal{M},w}$ as a specification of the doubly exponentially many tape cells needed for encoding the ATM.

Beyond tape cells, additional elements of an interpretation domain of $\Sigma_{\mathcal{M},w}$ represent configurations of $\mathcal{M}$ that are described using further signature symbols:

- *init*: constant for the initial configuration of the ATM,
- $state_q(v)$ for $q \in Q$: in the configuration $v$, the ATM is in state $q$,
- $head(v, x)$ : in the configuration $v$, the ATM is located at the tape cell $x$,

**(1) Initialisation** (with $w = \alpha_0 \dots \alpha_n$)**:**

$$min_k(x_0) \wedge \bigwedge_{0 \le i \le n} succ_k(x_i, x_{i+1}) \rightarrow state_{q_0}(init) \wedge \bigwedge_{0 \le i \le n} symbol_{\alpha_i}(init, x_i) \wedge symbol_{\square}(init, x_{n+1})$$

$$symbol_{\square}(init, x) \wedge succ_k(x, y) \rightarrow symbol_{\square}(init, y)$$

**(2) Left and right transition rules** (for $\delta_r = \langle q, \alpha, q', \alpha', r\rangle$ and $\delta_l = \langle q, \alpha, q', \alpha', l\rangle$)**:**

$$state_q(v) \wedge head(v, x) \wedge symbol_\alpha(v, x) \wedge succ_k(x, y) \rightarrow \exists v'.\, next_{\delta_r}(v, v') \wedge state_{q'}(v') \wedge head(v', y) \wedge symbol_{\alpha'}(v', x)$$

$$state_q(v) \wedge head(v, x) \wedge symbol_\alpha(v, x) \wedge succ_k(y, x) \rightarrow \exists v'.\, next_{\delta_l}(v, v') \wedge state_{q'}(v') \wedge head(v', y) \wedge symbol_{\alpha'}(v', x)$$

**(3) Inertia:**

$$next_\delta(v, v') \wedge head(v, x) \wedge succt(x, y) \wedge symbol_\alpha(v, y) \rightarrow symbol_\alpha(v', y)$$

$$next_\delta(v, v') \wedge head(v, x) \wedge succt(y, x) \wedge symbol_\alpha(v, y) \rightarrow symbol_\alpha(v', y)$$

**(4) Existential Acceptance** (for $q \in E$)**:**

$$state_q(v) \wedge next_\delta(v, v') \wedge accept(v') \rightarrow accept(v)$$

**(5) Universal Acceptance** (for $q \in U$, $\tilde{\Delta} = \{\langle q, \alpha, q', \alpha', d\rangle \in \Delta \mid d \in \{l, r\}\}$)**:**

$$state_q(v) \wedge head(v, x) \wedge symbol_\alpha(v, x) \wedge \bigwedge_{\delta \in \tilde{\Delta}} (next_\delta(v, v_\delta) \wedge accept(v_\delta)) \rightarrow accept(v)$$

Rules are instantiated for all $q, q' \in Q$, $\alpha, \alpha' \in \mathscr{A}$, and $\delta \in \Delta$.

**Fig. 1.** Rule set $\Sigma^{\text{exec}}_{\mathcal{M},w}$ simulating initialisation and execution on an ATM

- $symbol_\alpha(v, x)$ with $\alpha \in \mathscr{A}$: in configuration $v$, tape cell $x$ contains symbol $\alpha$,
- $accept(v)$: the ATM accepts configuration $v$.

Furthermore, for every $\delta \in \Delta$ we use $next_\delta(x, y)$ to indicate that configuration $x$ is changed to configuration $y$ by the transition $\delta$.

Consider the rule base $\Sigma_{\mathcal{M},w} = \Sigma^{\text{tape}}_{\mathcal{M},w} \cup \Sigma^{\text{exec}}_{\mathcal{M},w}$ with $\Sigma^{\text{exec}}_{\mathcal{M},w}$ as given in Fig. 1. It can be easily checked that it is glut-guarded, where the only glut variables are $v$, $v'$, and $v_\delta$ which represent ATM configurations. In particular, the predicates in $\Sigma^{\text{tape}}_{\mathcal{M},w}$ do not occur in rule heads of $\Sigma^{\text{exec}}_{\mathcal{M},w}$, and there are no cycles in the existential dependency graph of $\Sigma^{\text{tape}}_{\mathcal{M},w}$. Indeed, tape positions are encoded using the exponential effects (Theorem 4) of expanding non-glut variables, while ATM configurations are encoded using glut variables that are guarded.

Moreover, $\Sigma^{\text{exec}}_{\mathcal{M},w}$ realises the behaviour of the described ATM. As a peculiarity, note that acceptance is propagated backwards from the final accepting configurations. Inertia rules are used to copy the content of unchanged tape positions from one configuration to the next. We will show that checking whether the initial configuration is accepting is equivalent to checking whether the fact $accept(init)$ is a consequence of $\Sigma_{\mathcal{M},w}$.

We need to investigate the relationship between elements of an interpretation that satisfies $\Sigma_{\mathcal{M},w}$ and configurations of $\mathcal{M}$. Given an interpretation $\mathcal{I}$ of $\Sigma_{\mathcal{M},w}$, we say that an element $e$ of the domain of $\mathcal{I}$ *represents* a configuration $\alpha_1 \dots \alpha_{i-1} q \alpha_i \dots \alpha_m$ if $e \in state^{\mathcal{I}}_q$, $\langle e, d_i\rangle \in head^{\mathcal{I}}$, and, for every $j \in \{1, \dots, 2^{(2^k)}\}$, $\langle e, d_j\rangle \in symbol^{\mathcal{I}}_\alpha$ whenever (i) $j \le m$ and $\alpha = \alpha_m$, or (ii) $j > m$ and $\alpha = \square$.

Given some model $\mathcal{I}$ of $\Sigma_{\mathcal{M},w}$, we will now show that if some element $e$ of $\mathcal{I}$ represents a configuration $\theta$ and some transition $\delta$ is applicable to $\theta$, then there exists a domain element $e'$ with $\langle e, e'\rangle \in \mathit{next}_\delta^{\mathcal{I}}$ that represents the result of applying $\delta$ to $\theta$. To see this consider an element $e$, configuration $\theta$, and transition $\delta$ as in the claim. Then one of the axioms (2) applies, and $e$ must also have an according successor $e'$. This successor represents the correct state, head position, and symbol at head position $i$ of $e$, again by the axioms (2). By axioms (3), symbols at all other positions are also correctly transferred from $e$ to $e'$.

We can now show that a word $w$ is accepted by $\mathcal{M}$ iff $\mathit{accept}(\mathit{init})$ is a consequence of $\Sigma_{\mathcal{M},w}$.

We first show that for an arbitrary model $\mathcal{I}$ of $\Sigma_{\mathcal{M},w}$, any element $e$ of $\mathcal{I}$ that represents an accepting configuration $\theta$ satisfies $e \in \mathit{accept}^{\mathcal{I}}$.

We use an inductive argument along the recursive definition of acceptance. If $\theta$ is a universal configuration then all successors of $\theta$ are accepting, too. By our previous argument, for any $\delta$-successor $\theta'$ of $\theta$ there is a corresponding $e'$ with with $\langle e, e'\rangle \in \mathit{next}_\delta^{\mathcal{I}}$. By the induction hypothesis for $\theta'$, $e'$ is in $\mathit{accept}^{\mathcal{I}}$. Since this holds for all $\delta$-successors of $\theta$, axiom (5) implies $e \in \mathit{accept}^{\mathcal{I}}$. Especially, this argument covers the base case where $\theta$ has no successors.

If $\theta$ is an existential configuration, then there is some accepting $\delta$-successor $\theta'$ of $\theta$. Again by the previous argument, we have $\langle e, e'\rangle \in \mathit{next}_\delta^{\mathcal{I}}$ for an $e'$ that represents $\theta'$, and $e' \in \mathit{accept}^{\mathcal{I}}$ by the induction hypothesis. Hence axiom (4) applies and also concludes $e \in \mathit{accept}^{\mathcal{I}}$.

Since $\mathit{init}^{\mathcal{I}}$ represents the initial configuration of the ATM due to rules (1), this shows that $\mathit{init}^{\mathcal{I}} \in \mathit{accept}^{\mathcal{I}}$ whenever the initial configuration is accepting.

It remains to show the converse: if the initial configuration is not accepting, there is some interpretation $\mathcal{I}$ such that $a^{\mathcal{I}} \notin \mathit{accept}^{\mathcal{I}}$. To this end, we define a canonical interpretation $\mathcal{K}$ of $\Sigma_{\mathcal{M},w}$ as follows. The domain of $\mathcal{K}$ is the union of one set $D_{\text{cells}} = \{d_1, \ldots, d_{2^{(2^k)}}\}$ encoding the $2^{(2^k)}$ tape cells of $\mathcal{M}$ and another set $D_{\text{conf}}$ containing configurations of $\mathcal{M}$ that have size $2^{(2^k)} + 1$ (i.e. that encode a tape of length $2^{(2^k)}$, possibly with trailing blanks). The predicates from $\Sigma_{\mathcal{M},w}^{\text{tape}}$ are interpreted such that $\langle d_j, d_{j+1}\rangle \in \mathit{succ}_k{}^{\mathcal{I}}$ and $\langle d_j, d_{j'}\rangle \in \mathit{succt}^{\mathcal{I}}$ for all $j, j' \in \{1, \ldots, 2^{(2^k)}\}$ with $j < j'$ as well as $\mathit{min}_k^{\mathcal{I}} = \{d_1\}$ and $\mathit{max}_k^{\mathcal{I}} = \{d_{2^{(2^k)}}\}$; the other predicates are chosen such that all rules from $\Sigma_{\mathcal{M},w}^{\text{tape}}$ are satisfied as discussed above. The interpretations for the predicates $\mathit{state}_q$, $\mathit{head}$, and $\mathit{symbol}_\alpha$ are defined as expected so that every configuration represents itself but no other configuration. Especially, $\mathit{init}^{\mathcal{K}}$ is the initial configuration. Given two configurations $\theta$ and $\theta'$, and a transition $\delta$, we define $\langle \theta, \theta'\rangle \in \mathit{next}_\delta^{\mathcal{K}}$ iff there is a transition $\delta$ from $\theta$ to $\theta'$. Finally, $\mathit{accept}^{\mathcal{K}}$ is defined to be the set of accepting configurations.

By checking the individual axioms of Fig. 1, it is easy to see that $\mathcal{K}$ also satisfies $\Sigma_{\mathcal{M},w}^{\text{exec}}$. Now if the initial configuration is not accepting, $\mathit{init}^{\mathcal{K}} \notin \mathit{accept}^{\mathcal{K}}$ by construction. Thus $\mathcal{K}$ is a counterexample for $\mathit{accept}(\mathit{init})$ which thus is not a logical consequence.

□

## 7 Conclusion

We have extended the notions of weak acyclicity and weak (frontier-)guardedness, introduced a versatile new method for eliminating existential quantifiers, and applied these insights to define glut-frontier-guarded rules as one of the most expressive known existential rule languages for which query answering is decidable. Yet, a wide range of open issues still needs to be tackled for developing both the foundations of the field and applications to use these novel approaches.

Some immediate questions raised by this work concern the query complexity for fixed non-ground rules (data complexity) or for fixed signatures (bounded arity). A concurrent anonymous submission to this conference addresses these issues for previously defined rule languages, and it will be interesting to lift the respective methods to our cases.

More generally, further efforts are needed to continue the consolidation of rule languages that was started herein. To this end, modular reduction techniques for simplifying rule sets can be of great utility for advancing towards a unified theory of decidable existential rules.

## References


1. Andréka, H., Németi, I., van Benthem, J.: Modal languages and bounded fragments of predicate logic. J. of Philosophical Logic 27(3), 217–274 (1998)
2. Baget, J.F., Leclère, M., Mugnier, M.L.: Walking the decidability line for rules with existential variables. In: Lin, F., Sattler, U., Truszczynski, M. (eds.) Proc. 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'10). pp. 466–476. AAAI Press (2010)
3. Baget, J.F., Leclère, M., Mugnier, M.L., Salvat, E.: Extending decidable cases for rules with existential variables. In: Boutilier, C. (ed.) Proc. 21st Int. Conf. on Artificial Intelligence (IJCAI'09). pp. 677–682. IJCAI (2009)
4. Bárány, V., Gottlob, G., Otto, M.: Querying the guarded fragment. In: Proc. 25th Annual IEEE Symposium on Logic in Computer Science (LICS'10). pp. 1–10. IEEE Computer Society (2010)
5. Beeri, C., Vardi, M.Y.: The implication problem for data dependencies. In: Even, S., Kariv, O. (eds.) Proc. 8th Colloquium on Automata, Languages and Programming (ICALP'81). LNCS, vol. 115, pp. 73–85. Springer (1981)
6. Calì, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. In: Brewka, G., Lang, J. (eds.) Proc. 11th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'08). pp. 70–80. AAAI Press (2008)
7. Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. In: Paredaens, J., Su, J. (eds.) Proc. 28th Symposium on Principles of Database Systems (PODS'09). pp. 77–86. ACM (2009)
8. Calì, A., Gottlob, G., Pieris, A.: Advanced processing for ontological queries. Proceedings of VLDB 2010 3(1), 554–565 (2010)
9. Calì, A., Gottlob, G., Pieris, A.: Query answering under non-guarded rules in Datalog+/-. In: Hitzler, P., Lukasiewicz, T. (eds.) Proc. 4th Int. Conf. on Web Reasoning and Rule Systems (RR 2010). LNCS, vol. 6333, pp. 1–17. Springer (2010)

10. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. of Automated Reasoning 39(3), 385–429 (2007)
11. Chandra, A.K., Kozen, D.C., Stockmeyer, L.J.: Alternation. J. of the ACM 28(1), 114–133 (1981)
12. Chandra, A.K., Lewis, H.R., Makowsky, J.A.: Embedded implicational dependencies and their inference problem. In: Proc. 13th Annual ACM Symposium on Theory of Computation (STOC'81). pp. 342–354. ACM (1981)
13. Courcelle, B.: The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. Information and Computation 85(1), 12–75 (1990)
14. Deutsch, A., Tannen, V.: Reformulation of XML queries and constraints. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) Proc. 9th Int. Conf. on Database Theory (ICDT'03). LNCS, vol. 2572, pp. 225–241. Springer (2003)
15. Fagin, R., Kolaitis, P.G., Miller, R.J., Popa, L.: Data exchange: semantics and query answering. Theoretical Computer Science 336(1), 89–124 (2005)
16. Johnson, D.S., Klug, A.: Testing containment of conjunctive queries under functional and inclusion dependencies. In: Proc. 1st Symposium on Principles of Database Systems (PODS'82). pp. 164–169. ACM (1982)
17. Maier, D., Mendelzon, A.O., Sagiv, Y.: Testing implications of data dependencies. ACM Transactions on Database Systems 4, 455–469 (1979)
18. Patel-Schneider, P.F., Horrocks, I.: A comparison of two modelling paradigms in the Semantic Web. J. of Web Semantics 5, 240–250 (2007)