# Acyclicity Conditions and their Application to Query Answering in Description Logics

**Bernardo Cuenca Grau, Ian Horrocks, Markus Krötzsch, Clemens Kupke, Despoina Magka, Boris Motik,** and **Zhe Wang**
Department of Computer Science, Oxford University
Oxford, United Kingdom

**Abstract**

Answering conjunctive queries (CQs) over a set of facts extended with existential rules is a fundamental reasoning problem although undecidable due to non-termination of the main reasoning algorithm used—the *chase*. Several acyclicity conditions have been formulated that ensure chase termination. In this paper, we show that acyclicity can also be practically relevant for *description logic (DL)* reasoning. Due to the high complexity of answering CQs over DL ontologies, applications often solve this problem using *materialisation*, in which ontology consequences are precomputed using variants of the chase. Due to the non-termination problem, the execution of the algorithm is restricted only to rules that fall within the OWL 2 RL profile, which results in incomplete reasoning. After presenting two novel acyclicity conditions (*model-faithful acyclicity* (MFA) and *model-summarising acyclicity* (MSA)), we investigate the practical applicability of these and other acyclicity conditions for DL query answering. Our experiments reveal that many existing ontologies are MSA and that materialisation is typically not too large. Thus, our results suggest that principled, materialisation-based reasoning for ontologies beyond the OWL 2 RL profile may be practically feasible.

## Introduction

Existential rules are positive, function-free first-order implications that may contain existentially quantified variables in the head. In databases, they are known as *dependencies* (Abiteboul, Hull, and Vianu 1995) and are used to capture a wide range of schema constraints; they have, e.g., been used as declarative rules for data transformation in *data exchange*—the problem of transforming a database structured according to a source schema into a database structured according to a target schema (Fagin et al. 2005). They are also the basis for KR formalisms, such as datalog$^{\pm}$ (Calì, Gottlob, and Pieris 2010; Calì et al. 2010).

Answering *conjunctive queries* (CQs) over a set of facts extended with existential rules is a fundamental reasoning problem in both database and KR settings. This problem is undecidable (Beeri and Vardi 1981), and it can be characterised using *chase* (Johnson and Klug 1984; Maier, Mendelzon, and Sagiv 1979): a technique, closely related to hypertableau (Motik, Shearer, and Horrocks 2009), that computes facts implied by the rules in a forward-chaining manner, thus producing a *universal* model in which the query is evaluated.

Rules with existentially quantified variables in the head—which we call *generating rules*—may cause chase to generate new individuals, and cycles involving generating rules may lead to non-termination; moreover, determining whether chase terminates on a set of rules is undecidable. Chase, however, has been used to identify decidable classes of existential rules, and this has been done in two ways. In the first approach, rules are *restricted* such that their (possibly infinite) universal models can be represented using finitary means; this includes rules with universal models of bounded treewidth (Baget et al. 2011) and guarded rules (Calì et al. 2010). In the second approach, rules are *checked* using an *acyclicity* condition that is sufficient (but not necessary) to prove chase termination; roughly speaking, acyclicity conditions analyse information flow between the rules to ensure that no cyclic applications of generating rules are possible. *Weak acyclicity* (WA) (Fagin et al. 2005) was one of the first notions, and it was extended to *safety* (SF) (Meier, Schmidt, and Lausen 2009), *stratification* (ST) (Deutsch, Nash, and Remmel 2008), *acyclicity of a graph of rule dependencies* (aGRD) (Baget, Mugnier, and Thomazo 2011) *joint acyclicity* (JA) (Krötzsch and Rudolph 2011), and *super-weak acyclicity* (SWA) (Marnette 2009).

Acyclicity conditions are relevant for at least two reasons. First, unlike guarded rules, they do not restrict the shape of the structures that the rules can axiomatise; rather, they ensure that the rules can axiomatise only finite structures. Second, they ensure that the chase result can be stored and manipulated as if it were a database. This is important in data exchange, where the goal is to materialise the resulting database.

In this paper, we show that acyclicity is also relevant for *description logics (DLs)*, the KR formalisms underpinning the OWL 2 ontology language (Cuenca Grau et al. 2008). CQ answering is a key reasoning

service in many DL applications, which has been studied for many DLs (Calvanese et al. 2007; Krötzsch, Rudolph, and Hitzler 2007b; Glimm et al. 2008; Ortiz, Calvanese, and Eiter 2008; Lutz, Toman, and Wolter 2009; Pérez-Urbina, Motik, and Horrocks 2010; Rudolph and Glimm 2010; Kontchakov et al. 2011).

Due to the high complexity of answering CQs over expressive DLs, however, applications often solve this problem using *materialisation*, in which ontology consequences are precomputed using forward-chaining and stored in a semantic data store; examples include Oracle's Semantic Data Store, Sesame, Jena, OWLim, and DLE-Jena (Wu et al. 2008; Meditskos and Bassiliades 2008; Kiryakov, Ognyanov, and Manov 2005). This approach is possible if (*i*) the ontology is *Horn* (Hustadt, Motik, and Sattler 2005), and (*ii*) forward-chaining is guaranteed to terminate. In practice, condition (*ii*) is achieved by computing the materialisation using only inference rules corresponding to the part of the ontology that is in the OWL 2 RL profile; this excludes generating rules and so is terminating but incomplete in general. Even if generating rules are partially supported, as is the case in systems such as OWLim and Jena (Bishop and Bojanov 2011), this is typically rather ad hoc, does not guarantee completeness, and may even result in non-termination. Acyclicity conditions can be used to address these issues: if a Horn DL ontology is acyclic, a complete materialisation can be computed without the risk of non-termination.

Motivated by the practical importance of chase termination, we explore the landscape of acyclicity conditions, and present two novel conditions: *model-faithful acyclicity* (MFA) and *model-summarising acyclicity* (MSA). We then go investigate the practical applicability of these and other acyclicity conditions for query answering over DL ontologies.

Roughly speaking, our acyclicity conditions use a particular model of the rules to analyse the implications between existential quantifiers, which is why we call them *model based*. In particular, MFA uses the actual "canonical" model induced by the rules, which makes it a very general condition. We prove that checking whether a set of existential rules is MFA is 2ExpTime-complete, and it becomes ExpTime-complete if the predicates in the rules are of bounded arity. Due to the high complexity of MFA checking, MFA may be unsuitable for practical application, so we introduce MSA. Intuitively, MSA can be understood as MFA in which analysis is performed over models that "summarise" (or overestimate) the actual models. Checking MSA of existential rules can be realised via checking entailment of ground atoms in datalog programs; we use this close connection between MSA and datalog to prove that checking MSA is ExpTime-complete for general existential rules, and that it becomes coNP-complete if the arity of rule predicates is bounded; finally, we show that MSA is strictly more general than SWA—one of the most general acyclicity conditions currently is use.

Both of these conditions can be applied to general existential rules *without* equality. Equality can be incorporated via *singularisation* (Marnette 2009)—a technique that transforms the rules to encode the effects of equality. Singularisation is orthogonal to acyclicity: after computing the transformed rules, one can use MFA, MSA, or any acyclicty notion to check whether the result is acyclic; if so, chase with the original set of rules will terminate. Unfortunately, singularisation is nondeterministic: some ways of transforming the rules may produce acyclic rule sets, but not all ways will do so. Thus, we refine singularisation to obtain an upper and a lower bound for acyclicity. We also show that, when used with JA, the lower bound coincides with WA.

Finally, we turn our attention to the theoretical and practical questions of applying acyclicity to the problem of CQ answering over DLs. On the theoretical side, we show that checking MFA and MSA of Horn-$\mathcal{SHIQ}$ ontologies is PSpace- and PTime-complete, respectively, and that CQs can be answered over acyclic Horn-$\mathcal{SHIQ}$ ontologies in PSpace. The latter problem is ExpTime-hard for general (i.e., not acyclic) Horn-$\mathcal{SHIQ}$ ontologies (Ortiz, Rudolph, and Simkus 2011), so acyclicity makes the problem easier. Furthermore, Horn ontologies can be extended with arbitrary SWRL rules (Horrocks and Patel-Schneider 2004) without affecting neither decidability nor worst-case complexity, provided that the union of the ontology and SWRL rules is acyclic; this is in contrast to the general case, where SWRL extensions lead to undecidability.

On the practical side, we explore the limits of reasoning with acyclic OWL 2 ontologies via materialisation. We checked MFA, MSA, and JA of a library with 149 Horn ontologies; to estimate the impact of materialisation, we compared the size of the materialisation with the number of facts in the original ontologies. Our experiments revealed that many ontologies are MSA, and that some complex ones are MSA but not JA. Materialisation is typically not too large. Our results suggest that principled, materialisation-based reasoning for ontologies beyond the OWL 2 RL profile may be feasible. This paper is accompanied with an online technical report containing all proofs.[1]

## Preliminaries

We use the standard notions of constants, function symbols, and predicate symbols, where the latter two are associated with integer arity; $\approx$ is the equality predicate. Variables, terms, substitutions, atoms, and first-order formulae, sentences, interpretations (i.e., structures), and models are defined as usual. We abbreviate with $\vec{t}$ a vector of terms $t_1, \dots, t_n$ and define $|\vec{t}| = n$; with $\varphi(\vec{x})$ we stress that $\vec{x} = x_1, \dots, x_n$ are the free variables of a formula $\varphi$, and $\varphi\sigma$ is the result of applying a substitution $\sigma$ to $\varphi$. A term, atom, or formula is *ground* if it does not contain variables; a *fact* is a ground atom. The *depth* $\mathsf{dep}(t)$ of a term $t$ is defined as 0 if $t$ is a constant or a variable, and $\mathsf{dep}(t) = 1 + \max_{i=1}^{n} \mathsf{dep}(t_i)$ if

[1] http://www.cs.ox.ac.uk/isg/TR/acyclicity.pdf

$t = f(\vec{t})$. Satisfaction of a sentence $\varphi$ in an interpretation $I$ (written $I \models \varphi$), and entailment of a sentence $\psi$ from a sentence $\varphi$ (written $\varphi \models \psi$), are defined as usual. By a slight abuse of notation, we identify a conjunction of atoms with a set of atoms. A term $t'$ is a *subterm* of a term $t$ if $t' = t$ or $t = f(\vec{t})$ and $t'$ is a subterm of some $t_i \in \vec{t}$; if additionally $t' \neq t$, then $t'$ is a *proper* subterm of $t$. An atom $P(\vec{t})$ *contains* a term $t$ if $t \in \vec{t}$, and a set of atoms $I$ *contains* $t$ if some atom in $I$ contains $t$.

An *instance* is a finite set of function-free facts. An *existential rule* (or just *rule*) is a sentence of the form

$$\forall \vec{x} \forall \vec{z}.[\varphi(\vec{x}, \vec{z}) \rightarrow \exists \vec{y}.\psi(\vec{x}, \vec{y})] \tag{1}$$

where $\varphi(\vec{x}, \vec{z})$ and $\psi(\vec{x}, \vec{y})$ are conjunctions of atoms, and $\vec{x}$, $\vec{y}$, and $\vec{z}$ are pair-wise disjoint. Formula $\varphi$ is the *body*, formula $\psi$ is the *head*, and quantifiers $\forall \vec{x} \forall \vec{z}$ are often omitted. If $\vec{y}$ is empty, the rule is *datalog*. In database theory, satisfaction and entailment are often considered only w.r.t. *finite* interpretations under the *unique name assumption* (UNA), where distinct constants are interpreted as distinct elements; in contrast, such assumptions are not customary in KR. Since we study rules that can be satisfied in finite models, the restriction to finite satisfiability is immaterial; also, we do not assume UNA, which can be axiomatised if needed.

A *conjunctive query* (CQ) is a formula $Q(\vec{x})$ of the form $\exists \vec{y}.\varphi(\vec{x}, \vec{y})$, where $\varphi(\vec{x}, \vec{y})$ is a conjunction of atoms; the query is *Boolean* if $\vec{x}$ is empty. A substitution $\theta$ mapping $\vec{x}$ to constants is an *answer* to $Q(\vec{x})$ w.r.t. a set of rules $\Sigma$ and instance $I$ if $\Sigma \cup I \models Q(\vec{x})\theta$.

In first-order logic, $\approx$ is commonly assumed to have a predefined interpretation. The semantics of $\approx$, however, can be axiomatised explicitly. Let $\Sigma$ be a set of rules; w.l.o.g. we assume that no rule in $\Sigma$ contains $\approx$ in the body. Then, $\Sigma_\approx = \emptyset$ if $\approx$ does not occur in $\Sigma$; otherwise, $\Sigma_\approx$ contains the following rules:

$$\rightarrow x \approx x \tag{2}$$

$$x_1 \approx x_2 \rightarrow x_2 \approx x_1 \tag{3}$$

$$x_1 \approx x_2 \wedge x_2 \approx x_3 \rightarrow x_1 \approx x_3 \tag{4}$$

$$P(\vec{x}) \wedge x_i \approx x_i' \rightarrow P(x_1, \ldots, x_i', \ldots, x_n) \tag{5}$$

The consequences of $\Sigma$ (where $\approx$ is treated as having a well-defined interpretation) and $\Sigma \cup \Sigma_\approx$ (where $\approx$ is treated as an ordinary predicate) coincide.

Sometimes we use *skolemisation* to interpret rules in *Herbrand* interpretations—possibly infinite sets of ground atoms. In particular, the skolemisation of an existential rule $r$ of the form (1) is the rule

$$\varphi(\vec{x}, \vec{z}) \rightarrow \psi(\vec{x}, \vec{y})\theta \tag{6}$$

where for each $y_i \in \vec{y}$ we have $\theta(y_i) = f_r^i(\vec{x})$ with $f_r^i$ a function symbol globally unique for $r$ and $y_i$. The skolemisation $\mathsf{sk}(\Sigma)$ of a set of rules $\Sigma$ is obtained by skolemising each rule in $\Sigma$. For each CQ $Q(\vec{x})$, instance $I$, and substitution $\sigma$, we have $\Sigma \cup I \models Q(\vec{x})\sigma$ if and only if $\mathsf{sk}(\Sigma) \cup \Sigma_\approx \cup I \models Q(\vec{x})\sigma$.

Answering CQs can be characterised using *chase*, and we use the *skolem chase* variant (Marnette 2009). The result of *applying* a skolemised rule $r = \varphi \rightarrow \psi$ to a set of ground atoms $I$ is the smallest set $r(I)$ that contains $\psi\sigma$ for each substitution $\sigma$ from variables in $r$ to terms in $I$ such that $\varphi\sigma \subseteq I$; furthermore, for $\Omega$ a set of rules, $\Omega(I) = \bigcup_{r \in \Omega} r(I)$. Let $I$ be a finite set of ground atoms, and let $\Sigma$ be a set of rules. Let $\Sigma' = \mathsf{sk}(\Sigma) \cup \Sigma_\approx$, and let $\Sigma'_f$ and $\Sigma'_n$ be the subsets of $\Sigma'$ containing rules with and without function symbols, respectively. The *chase sequence* for $I$ and $\Sigma$ is a sequence of sets of facts $I^0_\Sigma, I^1_\Sigma, \ldots$ where $I^0_\Sigma = I$, and $I^i_\Sigma$ for $i > 0$ is as follows:

- if $\Sigma'_n(I^{i-1}_\Sigma) \not\subseteq I^{i-1}_\Sigma$, then $I^i_\Sigma = I^{i-1}_\Sigma \cup \Sigma'_n(I^{i-1}_\Sigma)$,
- otherwise $I^i_\Sigma = I^{i-1}_\Sigma \cup \Sigma'_f(I^{i-1}_\Sigma)$.

The *chase* of $I$ and $\Sigma$ is defined as $I^\infty_\Sigma = \bigcup_i I^i_\Sigma$; note that $I^\infty_\Sigma$ can be infinite. Chase can be used as a 'database' for answering CQs: a substitution $\sigma$ is an answer to $Q$ over $\Sigma$ and $I$ iff $I^\infty_\Sigma \models Q\sigma$. Chase of $I$ and $\Sigma$ *terminates* if $i \geq 0$ exists such that $I^i_\Sigma = I^j_\Sigma$ for each $j \geq i$; chase of $\Sigma$ terminates *universally* if the chase of $I$ and $\Sigma$ terminates for each $I$. If the skolem chase of $I$ and $\Sigma$ terminates, then the nonoblivious chase (Fagin et al. 2005), and the core chase (Deutsch, Nash, and Remmel 2008) of $I$ and $\Sigma$ terminate as well; hence, our results are applicable to all these chase variants.

The *critical instance* $I^*_\Sigma$ for rules $\Sigma$ contains all facts constructed using all predicates in $\Sigma$, all constants in the body of a rule in $\Sigma$, and a fresh constant $*$. If the skolem chase for $I^*_\Sigma$ and $\Sigma$ terminates, then the skolem chase of $\Sigma$ terminates universally (Marnette 2009).

Universal chase termination is undecidable, and various sufficient *acyclicity* conditions have been proposed. In the following, let $\Sigma$ be a set of rules where w.l.o.g. no variable occurs in more than one rule. A *position* is an expression of the form $P|_i$ where $P$ is an $n$-ary predicate and $1 \leq i \leq n$. Given a rule $r$ of the form (1) and a variable $w$, the set $\mathsf{Pos}_B(w)$ of *body positions* of $w$ consists of all positions $P|_i$ for which $P(t_1, \ldots, t_n) \in \varphi(\vec{x}, \vec{z})$ exists with $t_i = w$. The set $\mathsf{Pos}_H(w)$ is defined analogously.

*Weak acyclicity* (WA) was proposed by Fagin et al. (2005), and it is applicable to rules with $\approx$. The *WA dependency graph* $D_\Sigma$ for $\Sigma$ is defined as follows. Vertices of $D_\Sigma$ are positions. Graph $D_\Sigma$ contains, for each $r \in \Sigma$ of the form (1), each variable $x$, and each $P|_i \in \mathsf{Pos}_B(x)$, a *regular edge* from $P|_i$ to each $Q|_j \in \mathsf{Pos}_H(x)$ such that $Q \neq \approx$ and, for each $y \in \vec{y}$ and each $Q|_j \in \mathsf{Pos}_H(y)$ such that $Q \neq \approx$, a *special edge* from $P|_i$ to $Q|_j$. Set $\Sigma$ is WA if $D_\Sigma$ does not contain a cycle going through a special edge. Atoms involving equality are effectively ignored by WA.

*Joint acyclicity* (JA) is a generalisation of WA (Krötzsch and Rudolph 2011) applicable to equality-free rules. For an existentially quantified variable $y$ in $\Sigma$, let $\mathsf{Move}(y)$ be the smallest set of positions such that $\mathsf{Pos}_H(y) \subseteq \mathsf{Move}(y)$, and $\mathsf{Pos}_H(x) \subseteq \mathsf{Move}(y)$ for each universally quantified variable $x$ with $\mathsf{Pos}_B(x) \subseteq \mathsf{Move}(y)$. The *JA dependency graph* of $\Sigma$ is as follows. The vertices are the existentially quantified variables in $\Sigma$. The graph has an edge from each

$y$ to each $y'$ such that the rule in which $y'$ occurs also contains a universally quantified variable $x$ such that $\mathsf{Pos}_B(x) \subseteq \mathsf{Move}(y)$ and $\mathsf{Pos}_H(x) \neq \emptyset$. Set $\Sigma$ is JA if its JA dependency graph does not contain a cycle.

*Super-weak acyclicity* (SWA) (Marnette 2009) is more general than JA on rules in which a variable occurs more than once in a body atom. Since such rules are not obtained from DL knowledge bases, we omit the somewhat technical definition of SWA.

Spezzano and Greco (2010) suggest a rule rewriting framework for chase termination. A set of rules $\Sigma$ is rewritten into a set of rules $\Sigma'$ and, subsequently, $\Sigma'$ is checked for some existing notions of acyclicity (e.g. weak acyclicity). Since this rewriting technique is orthogonal to the aforementioned termination conditions and can be combined with any acyclicity condition we do not examine it in more detail.

## Model-Faithful Acyclicity

Ensuring (universal) chase termination can often be beneficial, and even necessary if the chase result is to be physically stored and manipulated as a database. Conditions such as JA, however, do not guarantee chase termination on certain commonly occurring rules.

**Example 1.** *Let $\Sigma$ be the set of rules (7)–(9).*

$$r_1 = \quad A(x) \rightarrow \exists y_1.R(x, y_1) \wedge B(y_1) \quad (7)$$

$$r_2 = \quad R(x, z) \wedge B(z) \rightarrow A(x) \quad (8)$$

$$r_3 = \quad B(x) \rightarrow \exists y_2.R(x, y_2) \wedge C(y_2) \quad (9)$$

*Note that $\mathsf{Move}(y_1) = \{R|_2,\ B|_1,\ R|_1,\ A|_1\}$; hence, the JA dependency graph has a cyclic edge from $y_1$ to itself. Chase of $\Sigma$, however, terminates universally: Assume that $f$ and $g$ are used to skolemize $r_1$ and $r_3$. Given a fact $A(a)$, rule $r_1$ derives $R(a, f(a))$ and $B(f(a))$, and rule $r_3$ derives $R(f(a), g(f(a)))$ and $C(g(f(a)))$; after this, rule $r_2$ is not applicable to $R(f(a), g(f(a)))$ since variable $z$ in $r_2$ cannot be matched to $B(g(f(a)))$, and so the chase terminates.*

Note that rules $r_1$ and $r_2$ encode the DL axiom $A \equiv \exists R.B$, and rule $r_3$ encodes $B \sqsubseteq \exists R.C$; such axioms abound in OWL ontologies. To enable applications of chase termination outlined in the introduction, we next propose a less restrictive acyclicity condition.

Acyclicity conditions try to estimate whether applying chase to a rule can produce facts that can (possibly by applying chase to other rules) repeatedly trigger the same rule. The key difference between various conditions is how rule applicability is determined. For example, JA and SWA consider each variable in a rule in isolation and do not check satisfaction of all body atoms at once; hence, they overestimate rule applicability. For example, rule (8) is not applicable to the facts generated by rule (9), but this can be determined only by considering variables $x$ and $z$ in rule (8) simultaneously. More precise chase termination guarantees can be obtained by tracking rule applicability more 'faithfully'.

A simple solution is to be completely precise about rule applicability: one can run the skolem chase and then use sufficient checks to identify cyclic computations. Clearly, no sufficient, necessary, and computable condition for the latter can be given, so we must adopt a practical approach; for example, we can 'raise the alarm' and stop the run if the chase derives a term $f(\vec{t})$ where $f$ occurs in $\vec{t}$. This condition can be further refined; for example, one could stop only if $f$ occurs nested in a term some fixed number of times. The choice of the appropriate condition is thus application dependent; however, as our experiments show, checking only for one level of nesting suffices in many cases. In particular, no term $f(\vec{t})$ with $f$ occurring in $\vec{t}$ is generated when running chase of $\Sigma$ in Example 1.

Meier, Schmidt, and Lausen (2009) proposed a related idea, where the chase is extended to keep track of a monitor graph, which is used to track rule dependencies and then to stop the chase if certain conditions are satisfied. This approach uses a variant of the chase that is don't-know nondeterministic: while all possible chase applications produce a model, not all applications will terminate, which can make acyclicity checking difficult.

In contrast, our notion of acyclicity is independent from any concrete notion of chase. The given rules $\Sigma$ are transformed into a new set of rules $\Sigma'$, which tracks rule dependencies using fresh predicates; then, $\Sigma$ is identified as being acyclic if $\Sigma'$ entails a special nullary predicate $\mathsf{C}$. Since acyclicity is defined via entailment, it can be decided using any sound and complete theorem proving procedure for existential rules. Acyclicity guarantees termination of skolem chase, which then guarantees termination of nonoblivious and core chase as well.

We call our new acyclicity notion *model-faithful acyclicity* because it estimates rule application precisely, by examining the actual model of $\Sigma$.

**Definition 2.** *For each rule $r = \varphi(\vec{x}, \vec{z}) \rightarrow \exists \vec{y}.\psi(\vec{x}, \vec{y})$ and each variable $y_i \in \vec{y}$, let $\mathsf{F}^i_r$ be a fresh unary predicate unique for $r$ and $y_i$; also, let $\mathsf{S}$ and $\mathsf{D}$ be fresh binary predicates and let $\mathsf{C}$ be a fresh nullary predicate. Then, $\mathsf{MFA}(r)$ is the following rule:*

$$\varphi(\vec{x}, \vec{z}) \rightarrow \exists \vec{y}.[\psi(\vec{x}, \vec{y}) \wedge \bigwedge_{y_i \in \vec{y}} [\mathsf{F}^i_r(y_i) \wedge \bigwedge_{x_j \in \vec{x}} \mathsf{S}(x_j, y_i)]]$$

*For $\Sigma$ a set of rules, $\mathsf{MFA}(\Sigma)$ is the smallest set that contains $\mathsf{MFA}(r)$ for each rule $r \in \Sigma$, rules (10)–(11), and rule (12) instantiated for each predicate $\mathsf{F}^i_r$:*

$$\mathsf{S}(x_1, x_2) \rightarrow \mathsf{D}(x_1, x_2) \quad (10)$$

$$\mathsf{D}(x_1, x_2) \wedge \mathsf{S}(x_2, x_3) \rightarrow \mathsf{D}(x_1, x_3) \quad (11)$$

$$\mathsf{F}^i_r(x_1) \wedge \mathsf{D}(x_1, x_2) \wedge \mathsf{F}^i_r(x_2) \rightarrow \mathsf{C} \quad (12)$$

*Set $\Sigma$ is* model-faithful acyclic *(MFA) w.r.t. an instance $I$ if $I \cup \mathsf{MFA}(\Sigma) \not\models \mathsf{C}$; furthermore, $\Sigma$ is* universally MFA[2] *if $\Sigma$ is MFA w.r.t. $I^*_\Sigma$.*

MFA is defined as a semantic, rather than a syntactic condition, and entailment $I \cup \mathsf{MFA}(\Sigma) \not\models \mathsf{C}$ can be checked using sound and complete first-order calculus.

[2] In the rest of this paper we typically omit 'universally'.

In the following section we show that MFA is strictly more general than SWA. We next show that MFA characterises derivations of skolem chase in a particular way.

**Definition 3.** *A term $t$ is* cyclic *if some $f(\vec{s})$ is a subterm of $t$, and some $f(\vec{u})$ is a proper subterm of $f(\vec{s})$.*

**Proposition 4.** *A set of rules $\Sigma$ is not MFA w.r.t. an instance $I$ iff $I^{\infty}_{\mathsf{MFA}(\Sigma)}$ contains a cyclic term.*

This characterisation immediately implies termination of skolem chase of MFA rules in 2ExpTime. The latter already holds if the rules are WA; hence, CQ answering for MFA rules is not harder than for WA.

**Proposition 5.** *If a set of rules $\Sigma$ is MFA w.r.t. an instance $I$, then the skolem chase for $I$ and $\Sigma$ terminates in double exponential time.*

Proposition 5 implies that answering a BCQ over MFA rules is in 2ExpTime; furthermore, Calì, Gottlob, and Pieris (2010) provide the matching lower bound for WA rules. We next prove that checking MFA w.r.t. a specific instance $I$ is in 2ExpTime, and that checking universal MFA is 2ExpTime-hard. These results provide tight bounds for both problems.

**Theorem 6.** *For $\Sigma$ a set of rules, deciding whether $\Sigma$ is MFA w.r.t. an instance $I$ is in* 2ExpTime*, and deciding whether $\Sigma$ is universally MFA is* 2ExpTime*-hard. Both results hold even if the arity of predicates in $\Sigma$ is bounded.*

The results of Theorem 6 are somewhat discouraging: acyclicity according to existing criteria can be checked in PTime or in NP. We consider MFA to be an "upper bound" of practically useful acyclicity conditions. We see two possibilities for improving these results. In the following section, we introduce an approximation of MFA that is easier to check; our evaluation shows that this condition often coincides with MFA in practice. In the rest of this section, we show that the complexity is lower for rules of the following shape.

**Definition 7.** *A rule $r$ of the form* (1) *is an* $\exists$-1 rule *if $\vec{y}$ is empty or $\vec{x}$ contains at most one variable.*

As discussed later on, $\exists$-1 rules capture (extensions of) Horn DLs. We next show that BCQ answering and MFA checking for $\exists$-1 rules is easier than for the general rules. The following theorem provides the upper bound; the lower bounds are given later on for a smaller class of rules that capture DLs.

**Theorem 8.** *Let $\Sigma$ be a set of $\exists$-1 rules, and let $I$ be an instance. Checking whether $\Sigma$ is MFA w.r.t. $I$ is in* ExpTime*. Furthermore, if $\Sigma$ is MFA, then answering a BCQ over $\Sigma$ and $I$ is in* ExpTime *as well.*

## Model-Summarising Acyclicity

The high cost of checking MFA of $\Sigma$ is due to the fact that the arity of function symbols in $\mathsf{sk}(\Sigma)$ is unbounded, and that the depth of cyclic terms can be linear in $\Sigma$. To obtain an acyclicity condition that is easier to check, we must coarsen the structure used for the analysis of cycles. Thus, we define *model-summarising acyclicity*, which "summarises" the models of $\Sigma$ by reusing the same constant to satisfy an existential quantifier, instead of introducing deeper terms.

**Definition 9.** *Let $\mathsf{S}$, $\mathsf{D}$, and $\mathsf{F}^i_r$ be as in Definition 2; furthermore, for each rule $r = \varphi(\vec{x},\vec{z}) \rightarrow \exists\vec{y}.\psi(\vec{x},\vec{y})$ and each variable $y_i \in \vec{y}$, let $\mathsf{c}^i_r$ be a fresh constant unique for $r$ and $y_i$. Then, $\mathsf{MSA}(r)$ is the following rule, where $\theta$ is a substitution that maps each variable $y_i \in \vec{y}$ to $\mathsf{c}^i_r$:*

$$\varphi(\vec{x},\vec{z}) \rightarrow \psi(\vec{x},\vec{y})\theta \wedge \bigwedge_{y_i \in \vec{y}} [\mathsf{F}^i_r(y_i)\theta \wedge \bigwedge_{x_j \in \vec{x}} \mathsf{S}(x_j, y_i)\theta]$$

*For $\Sigma$ a set of rules, $\mathsf{MSA}(\Sigma)$ is the smallest set that contains $\mathsf{MSA}(r)$ for each rule $r \in \Sigma$, rules* (10)–(11)*, and rule* (12) *instantiated for each predicate $\mathsf{F}^i_r$. Set $\Sigma$ is* model-summarising acyclic *(MSA) w.r.t. an instance $I$ if $I \cup \mathsf{MSA}(\Sigma) \not\models \mathsf{C}$; furthermore, $\Sigma$ is* universally MSA *if $\Sigma$ is MSA w.r.t. $I^*_\Sigma$.*

Note that $\mathsf{MSA}(\Sigma)$ is a set of datalog rules; hence, MSA can be checked using any datalog engine. This connection with datalog provides the complexity upper bound of checking MSA: the following theorem follows from the well known complexity results of checking entailment of a ground atom in a datalog program (Dantsin et al. 2001). The complexity of reasoning in datalog is $O(n^v)$ where $v$ is the max. number of variables in a rule and $n$ is the size of $\Sigma$; hence, we expect MSA checking to be feasible if the rules in $\Sigma$ are 'short'.

**Theorem 10.** *For $\Sigma$ a set of rules, deciding whether $\Sigma$ is MSA w.r.t. an instance $I$ is in* ExpTime*, and deciding whether $\Sigma$ is universally MSA is* ExpTime*-hard. The two problems are in co*NP *and co*NP*-hard, respectively, if the arity of the predicates in $\Sigma$ is bounded.*

We finish this section by proving strict inclusion relationships between MFA, MSA, and SWA. In particular, Theorem 11 and Example 12 show that that MFA is strictly more general than MSA.

**Theorem 11.** *If a set of rules $\Sigma$ is MSA (w.r.t. an instance $I$), then $\Sigma$ is MFA (w.r.t. $I$) as well.*

**Example 12.** *Let $\Sigma$ be the set of rules* (13)–(16)*.*

$$A(x) \rightarrow \exists y.R(x,y) \wedge B(y) \tag{13}$$
$$B(x) \rightarrow \exists y.S(x,y) \wedge T(y,x) \tag{14}$$
$$A(z) \wedge S(z,x) \rightarrow C(x) \tag{15}$$
$$C(z) \wedge T(z,x) \rightarrow A(x) \tag{16}$$

*$\Sigma$ is universally MFA, but not universally MSA.* $\Diamond$

We now show that MSA is more general than SWA, and thus also more general than JA. The converse does not hold: the set $\Sigma$ in Example 1 is MSA, but not SWA.

**Theorem 13.** *If a set of equality-free rules $\Sigma$ is SWA, then $\Sigma$ is universally MSA as well.*

## Handing Equality via Singularisation

JA and SWA can be applied to rules with equality provided that the rule set contains rules (2)–(5). In both

cases, however, rules (2) and (5) lead to a cycle as soon as the rule set contains an existential quantifier. MFA and MSA are slightly more robust but still fail to capture many practically relevant rule sets.

**Example 14.** *Consider the following set of rules.*

$$A(x) \wedge B(x) \rightarrow \exists y.[R(x,y) \wedge B(y)] \quad (17)$$

$$R(z,x_1) \wedge R(z,x_2) \rightarrow x_1 \approx x_2 \quad (18)$$

*On these rules and the critical instance, the skolem chase derives the following infinite set of facts.*

$$\begin{array}{cccc} R(*,f(*)) & B(f(*)) & * \approx f(*) & A(f(*)) \\ R(f(*),f(f(*))) & B(f(f(*))) & \ldots & \Diamond \end{array}$$

Example 14 shows that equalities between terms tend to proliferate during chase, which can lead to non-termination. Interestingly, the rules in the example are WA. This is because WA is sufficient for termination of the nonoblivious chase—a version of chase that expands an existential quantifier only if necessary. Already JA is more general than WA on rules without equality, so nonoblivious chase does not seem to provide an advantage over skolem chase w.r.t. termination on such rules; however, Example 14 shows that this is not the case for rules with equality.

Marnette (2009) proposed a solution to this problem based on a technique called *singularisation*. The idea is to only partially axiomatise $\approx$ as being reflexive, symmetric, and transitive, but without the replacement property cf. rule (5). A set of rules $\Sigma$ is modified in a way to take into account the lack of the replacement rules. This latter step is nondeterministic: there are many ways to modify $\Sigma$ and, while some modifications will lead to chase termination, not all will do so.

We recapitulate the definition of singularisation. A *marking* $M_r$ of a rule $r$ of the form (1) is a mapping from each $w \in \vec{x} \cup \vec{z}$ to a single occurrence of $w$ in $\varphi$; all other variable occurrences are *unmarked* and all constants are also unmarked. A marking $M$ of $\Sigma$ has exactly one marking $M_r$ for each $r \in \Sigma$. The *singularisation* of $\Sigma$ under $M$ is the set $\mathsf{Sing}(\Sigma, M)$ containing

- for each $r \in \Sigma$, a rule obtained by replacing each unmarked occurrence of a body term $t$ in $r$ with a fresh variable $z'$ and adding $t \approx z'$ to the body, and
- rules (2), (3), (4).

Note that $\mathsf{Sing}(\Sigma, M)$ is unique up to the renaming of the fresh variables. We write $\hat{x}$ to denote the marked occurrence of $x$ in a rule. The properties of singularisation can be summarised as follows: for an arbitrary set of rules $\Sigma$, a marking $M$ for $\Sigma$, an instance $I$, and a fact $P(\vec{c})$, we have $I \cup \Sigma \cup \Sigma_{\approx} \models P(\vec{c})$ if and only if

$$I \cup \mathsf{Sing}(\Sigma, M) \models \exists \vec{y}.[P(\vec{y}) \wedge \bigwedge_{y_i \in \vec{y}} y_i \approx c_i].$$

**Example 15.** *Singularisation of the marked rule* (19) *produces rule* (20)*. Note that singularisation is applied 'globally' to all rules, even to those without equality.*

$$A(\hat{x}) \wedge B(x) \wedge R(x,\hat{z}) \rightarrow C(x) \quad (19)$$

$$A(x) \wedge B(x_1) \wedge R(x_2,z) \wedge x \approx x_1 \wedge x \approx x_2 \rightarrow C(x) \quad (20)$$

The absence of rules (5) often allows the skolem chase to terminate on $\mathsf{Sing}(\Sigma, M)$; however, this may depend on the selected marking.

**Example 16.** *Rule* (17) *from Example 14 admits the following two markings:*

$$A(\hat{x}) \wedge B(x) \rightarrow \exists y.[R(x,y) \wedge B(y)] \quad (21)$$

$$A(x) \wedge B(\hat{x}) \rightarrow \exists y.[R(x,y) \wedge B(y)] \quad (22)$$

*The skolem chase does not universally terminate for the singularisation obtained from* (22)*; in contrast, the singularisation obtained from* (21) *is JA.* $\Diamond$

We use $\text{MFA}^{\exists}$ and $\text{MFA}^{\forall}$ to denote the classes of rule sets that are in MFA for *some* singularisation and for *all* singularisations, respectively; notions $\text{MSA}^{\exists}$, $\text{MSA}^{\forall}$, $\text{JA}^{\exists}$, and $\text{JA}^{\forall}$ are defined analogously. Clearly, $X^{\forall} \subseteq X^{\exists}$ for each $X \in \{\text{MFA}, \text{MSA}, \text{JA}\}$, and Example 16 shows this inclusion to be proper.

**Theorem 17.** *$JA^{\forall} = WA$.*

Checking all possible markings may not be feasible (there are exponentially many in the total number of variables occurring more than once in a rule body). Theorem 17 shows that $\text{JA}^{\forall}$ can be decided using WA. For the other cases, the following lemma shows how to reduce the number of markings.

**Lemma 18.** *Let $M$ and $M'$ be markings for $\Sigma$ that agree on all variables that occur in both body and head, but not necessarily on the variables that occur only in the body of a rule. Then $\mathsf{Sing}(\Sigma, M)$ is JA/MSA/MFA if and only if $\mathsf{Sing}(\Sigma, M')$ is JA/MSA/MFA.*

Despite this optimisation, the number of markings to check is still exponential; hence, we next describe a useful approximation. Let $\mathsf{Sing}_{\cup}(\Sigma) = \bigcup_{M \in \mathcal{M}} \mathsf{Sing}(\Sigma, M)$, where $\mathcal{M}$ is a set of all markings for $\Sigma$ that agree on all variables occurring only in the body of a rule in $\Sigma$. By Lemma 18, it is irrelevant how the markings of body variables are defined in $\mathcal{M}$. Let $\text{MFA}^{\cup}$ be the class containing rule sets $\Sigma$ for which $\mathsf{Sing}_{\cup}(\Sigma)$ is in MFA; $\text{MSA}^{\cup}$ and $\text{JA}^{\cup}$ are defined analogously. As the following theorem shows, $\mathsf{Sing}_{\cup}(\Sigma)$ provides a 'lower bound' on the result attainable via singularisation.

**Theorem 19.** *For each $X \in \{MFA, MSA, JA\}$, we have that $X^{\cup} \subseteq X^{\forall}$. The size of $\bigcup_{M \in \mathcal{M}} \mathsf{Sing}(\Sigma, M)$ is exponential in the maximal number of variables that occur more than once in the body of any one rule in $\Sigma$, and it is linear in the number of rules in $\Sigma$.*

This result is of particular interest when dealing with rules that are obtained from DLs, where each rule has at most one variable that occurs in the head as well as multiple times in the body. On such rule sets, the size of $\mathsf{Sing}_{\cup}(\Sigma)$ is linear in the size of $\Sigma$. For the general case, we can obtain the same complexity bounds despite the exponential increase in the number of rules:

**Theorem 20.** *Deciding whether $\Sigma$ is in $MFA^{\cup}$ ($MFA^{\exists}$, $MFA^{\forall}$) is* 2ExpTime*-complete. Deciding whether $\Sigma$ is in $MSA^{\cup}$ ($MSA^{\exists}$, $MSA^{\forall}$) is* ExpTime*-complete.*

## Acyclicity of DL Ontologies

We now turn our attention to applying acyclicity conditions to DL ontologies. DLs are KR formalisms that underpin the Web Ontology Language (OWL). DL ontologies are constructed from *atomic concepts* (i.e., unary predicates), *atomic roles* (i.e., binary predicates), and *individuals* (i.e., constants). Special atomic concepts $\top$ and $\bot$ denote universal truth and falsehood, respectively. For $R$ an atomic role, $R^-$ is an *inverse role*; inverse roles can be used in atoms: $R^-(t_1, t_2)$ is an abbreviation for $R(t_2, t_1)$. A *role* is an atomic or an inverse role. DLs provide a rich set of *constructors* for building *concepts* (first-order formulae with one free variable) from atomic concepts and roles. DL ontologies consist of *axioms* about concepts and roles; these correspond to first-order sentences. For simplicity, we consider only normalised ontologies, in which concepts are not nested. This is w.l.o.g., as each ontology can be normalised using a linear algorithm, and the normalised ontology is a conservative extension of the original one. In this paper, we consider only *Horn* DLs; ontologies in such DLs have at most one minimal Herbrand model, which is a prerequisite for materialisation-based reasoning—the main motivation for applying acyclicity to DLs.

A normalised Horn-$\mathcal{SRIQ}$ *TBox* $\mathcal{T}$ consists of axioms shown in the left-hand side of Table 1; in the table, $A$, $B$, and $C$ are atomic concepts (including possibly $\top$ and $\bot$), $R$, $S$, $T$ are (not necessarily atomic) roles, and $n$ is a positive integer. To guarantee decidability of reasoning, $\mathcal{T}$ must satisfy certain *global* conditions (Kutz, Horrocks, and Sattler 2006), which we omit for brevity. Roughly speaking, only so-called *simple* roles are allowed to occur in axioms of Type 2, and axioms of Type 6 must be *regular* according to a particular condition; the latter condition ensures that axioms of Type 6 can be represented as a nondeterministic finite automaton. Apart from Horn-$\mathcal{SRIQ}$, we also consider Horn-$\mathcal{SRI}$ TBoxes, which do not contain rules of Type 2, as well as Horn-$\mathcal{SHIQ}$ TBoxes, where $R = S = T$ in all rules of Type 6; all Horn-$\mathcal{SHIQ}$ TBoxes are regular.

Each Horn-$\mathcal{SRIQ}$ axiom corresponds to an existential rule as shown in Table 1. A minor difference is that axioms in Table 1 can contain $\bot$ in the head, which can make a TBox $\mathcal{T}$ unsatisfiable w.r.t. an instance $I$. This can be handled by considering $\bot$ to be just another atomic concept, without special meaning. Technically, this ensures that $I \cup \mathcal{T}$ is satisfiable in a model that can be constructed by skolem chase; however, we consider $I \cup \mathcal{T}$ to be unsatisfiable if $I \cup \mathcal{T} \models \exists y.\bot(y)$. We consider a substitution $\theta$ to be an answer to a CQ $Q(\vec{x})$ w.r.t. a $\mathcal{T}$ and $I$ if $I \cup \mathcal{T} \models \exists y.\bot(y)$ or $\mathcal{T} \cup I \models Q(\vec{x})\theta$. Due to this close correspondence between DL axioms and existential rules, in the rest of this paper we identify a TBox $\mathcal{T}$ with the corresponding set of rules.

We next investigate the complexity of BCQ answering over acyclic DL TBoxes. For the membership, note that all rules in Table 1 are $\exists$-1 rules; thus, Theorem 8 gives us an ExpTime upper bound. We next prove a matching lower bound for WA Horn-$\mathcal{SRI}$ rules. Intuitively, axioms of Type 6 allow us to axiomatise non-tree-like structures; although regularity ensures that axioms of Type 6 can be represented by a nondeterministic finite automaton, this automaton can be exponential, which may require one to examine all nodes in an exponential model of a Horn-$\mathcal{SRI}$ TBox. Furthermore, by Theorem 8, if we extend acyclic Horn-$\mathcal{SRIQ}$ rules $\Sigma_1$ with arbitrary SWRL rules $\Sigma_2$, reasoning stays ExpTime-complete, provided that $\Sigma_1 \cup \Sigma_2$ is acyclic; this is in contrast to general TBoxes for which SWRL extensions lead to undecidability. Thus, applications that need expressivity beyond what is available in OWL can benefit from the required expressivity without running into undecidability as long as the resulting ontology is acyclic.

**Theorem 21.** *Let $\mathcal{T}$ be a WA Horn-$\mathcal{SRI}$ TBox, let $I$ be an instance, and let $F$ be a fact. Then, checking whether $I \cup \mathcal{T} \models F$ is* ExpTime*-hard.*

Note that Theorem 21 applies to Horn-$\mathcal{SRI}$ and thus does not rely on a particular treatment of equality.

The proof of Theorem 21 can be adapted to obtain the lower bound for checking MFA of Horn-$\mathcal{SRI}$ rules.

**Proposition 22.** *Checking whether a Horn-$\mathcal{SRI}$ TBox is universally MFA is* ExpTime*-hard.*

As we show next, however, the complexity or query answering drops to PSpace for MFA Horn-$\mathcal{SHIQ}$ ontologies. In contrast, checking entailment of a single fact is ExpTime-hard in the general (i.e., not acyclic) case (Krötzsch, Rudolph, and Hitzler 2007a). This drop in complexity is due to the fact that, if $R = S = T$ in all rules of Type 6, the automaton describing roles is of polynomial size. Thus, although acyclic TBoxes can axiomatise existence of polynomially deep and exponentially large structures, these structures are tree-like, which allows us to explore the structures one path at a time using the well-known tracing technique. The main difficulty in the membership proof of the following theorem is due to the fact that queries can contain transitive roles, so one cannot roll a query up into a concept. Since the TBox is Horn, however, one can guess places in the model to which the query maps. Given one such guess, one can ground the query and check entailment of each ground query atom individually, while taking transitive roles into account. Furthermore, note that the proof of PSpace-hardness of concept satisfiability checking by Baader et al. (2007) is not applicable to Horn ontologies since it uses concepts with disjunctions.

**Theorem 23.** *Let $\mathcal{T}$ be Horn-$\mathcal{SHIQ}$ TBox, let $I$ be an instance such that $\mathcal{T}$ is MFA w.r.t. $I$, and let $Q$ be a BCQ. Then, deciding $I \cup \mathcal{T} \models Q$ is* PSpace*-complete.*

Although the proof of Theorem 23 takes into account ontology rules with equality (i.e., rules of Type 2), it assumes that equality is axiomatised by $\Sigma_\approx$ and hence it does not directly apply to *singularised* Horn-$\mathcal{SHIQ}$ rules. We conjuecture, however, that the result in the theorem holds regardless of singularisation.

The restriction to Horn-$\mathcal{SHIQ}$ rules also makes checking MFA easier. Roughly speaking, checking MFA

| | | |
|---|---|---|
| 1. | $A \sqsubseteq \exists R.B$ | $A(x) \rightarrow \exists y.[R(x,y) \wedge B(y)]$ |
| 2. | $A \sqsubseteq \leq 1\, R.B$ | $A(z) \wedge R(z,x_1) \wedge B(x_1) \wedge R(z,x_2) \wedge B(x_2) \rightarrow x_1 \approx x_2$ |
| 3. | $A \sqcap B \sqsubseteq C$ | $A(x) \wedge B(x) \rightarrow C(x)$ |
| 4. | $A \sqsubseteq \forall R.B$ | $A(z) \wedge R(z,x) \rightarrow B(x)$ |
| 5. | $R \sqsubseteq S$ | $R(x_1,x_2) \rightarrow S(x_1,x_2)$ |
| 6. | $R \circ S \sqsubseteq T$ | $R(x_1,z) \wedge S(z,x_2) \rightarrow T(x_1,x_2)$ |

Table 1: Axioms of normalised Horn-$\mathcal{SRIQ}$ ontologies and corresponding rules

w.r.t. an instance can be done by a minor variation of the query answering algorithm.

**Theorem 24.** *Let $\mathcal{T}$ be Horn-$\mathcal{SHIQ}$ TBox, and let $I$ be an instance. Then, deciding whether $\mathcal{T}$ is MFA w.r.t. $I$ is in* PSPACE*, and deciding whether $\mathcal{T}$ is universally MFA is* PSPACE*-hard.*

Finally, MSA provides us with a tractable condition for Horn-$\mathcal{SHIQ}$ rules. Intuitively, all rules in $\mathsf{MSA}(\mathcal{T})$ have a bounded number of variables and all predicates in $\mathsf{MSA}(\mathcal{T})$ are of bounded arity, which eliminates all sources of intractability in datalog reasoning.

**Theorem 25.** *Let $\mathcal{T}$ be Horn-$\mathcal{SHIQ}$ TBox, and let $I$ be an instance. Then, deciding whether $\mathcal{T}$ is MSA w.r.t. $I$ is in* PTIME*, and deciding whether $\mathcal{T}$ is universally MSA is* PTIME*-hard.*

This result also holds for singularised rules.

## Experiments

We have evaluated the applicability of various acyclicity conditions in practice. First, we implemented MFA, MSA, JA, and WA checkers, and used them to check acyclicity of a large corpus of Horn ontologies. Our goal was to determine whether a substantial portion of these ontologies are acyclic and could thus be used with (suitably extended) materialisation-based reasoners. Second, we computed the materialisation of the acyclic Horn ontologies and compared the size of the materialisation with the size of the original ABox. The goal of these tests was to see whether materialisation-based reasoning is practically feasible.

Tests were performed on the Oxford Super Computer HAL system with 8 2.8GHz processors and 16GB RAM. We used a repository of 149 OWL ontologies whose TBox axioms can be transformed into existential rules. These ontologies include many of those in the Gardiner corpus (Gardiner, Tsarkov, and Horrocks 2006), the LUBM ontology, and a number of ontologies from the Open Biomedical Ontology (OBO) corpus. All test ontologies are available online.[3]

### Acyclicity Tests

We implemented all acyclicity checks by adapting the HermiT reasoner. HermiT was used to transform an ontology into DL-clauses—formulae quite close to existential rules. DL-clauses were then preprocessed: at-least number restrictions in rule heads were replaced with existential quantification, atoms involving datatypes were eliminated, and DL-clauses with empty head were removed; datatypes and empty heads merely cause inconsistencies, and do not contribute to chase nontermination. If the DL-clauses contained equality, we check $X^{\cup}$ instead of $X$ for each $X \in \{\text{MFA}, \text{MSA}, \text{JA}\}$ as a 'lower-bound' for acyclicity. To obtain an 'upper bound' for acyclicity, we checked whether the ontology was already cyclic when ignoring the rules containing equality. These steps produced a set of existential rules, which were further modified as required to encode the desired ayclicity check. Finally, HermiT was used to test universal acyclicity of the ontology by checking logical entailment w.r.t. the critical instance.

| G-rules | Total | MSA | JA | WA |
|---|---|---|---|---|
| ontologies without equality | | | | |
| $< 100$ | 21 | 19 | 19 | 19 |
| 100–1K | 33 | 30 | 30 | 23 |
| 1K–5K | 18 | 14 | 14 | 12 |
| 5K–12K | 9 | 8 | 6 | 6 |
| 12K–160K | 7 | 5 | 3 | 3 |
| ontologies with equality | | | | |
| $< 100$ | 49 | 45 | 45 | 45 |
| 100–1K | 0 | 0 | 0 | 0 |
| 1K–5K | 2 | 0 | 0 | 0 |
| 5K–12K | 5 | 3 | 0 | 0 |
| 12K–160K | 5 | 0 | 0 | 0 |

Table 2: Results of acyclicity tests

Each acyclicity test was given a $500s$ timeout. The MSA test exceeded this limit on 2 ontologies, whereas the MFA test exceeded it on 26 ontologies. Of the 149 ontologies tested, 124 (83%) were MSA. Moreover, MFA and MSA are indistinguishable w.r.t. the test ontologies—that is, all MFA ontologies were found to be MSA as well (the converse holds per Theorem 11). Results are shown in Table 2. Given the large number of test ontologies, we cannot show results for each ontology. Instead, ontologies are grouped by number of generating rules (G-rules); for each group, the table shows the number of ontologies (Total) and the number of ontologies found to be MSA, JA, and WA.

Note that 7 large OBO ontologies were MSA but not JA; thus, MSA may be especially useful on large and

[3] `http://www.hermit-reasoner.com/2011/acyclicity/TestCorpus.zip`

| G-rules | Eq | DL | C | P | A |
|---|---|---|---|---|---|
| biological_process_xp_self.imports.owl | | | | | |
| 10980 | yes | $\mathcal{SRIF}$ | 22375 | 183 | 47454 |
| go_xp_regulation.owl | | | | | |
| 11187 | no | $\mathcal{SH}$ | 27883 | 5 | 50941 |
| biological_process_xp_cell.imports.owl | | | | | |
| 11274 | yes | $\mathcal{SRIF}$ | 24309 | 293 | 50386 |
| cellular_component_xp_go.imports.owl | | | | | |
| 11473 | no | $\mathcal{SR}$ | 35236 | 8 | 64026 |
| biological_. . . . _cellular_component.imports.owl | | | | | |
| 11798 | yes | $\mathcal{SRIF}$ | 25337 | 187 | 52759 |
| go_xp_regulation.imports.owl | | | | | |
| 23844 | no | $\mathcal{SR}$ | 34293 | 8 | 104473 |
| biological_. . . . _multi_organism_process.imports.owl | | | | | |
| 24678 | no | $\mathcal{SR}$ | 34410 | 21 | 104873 |

Table 3: MSA but not JA ontologies

complex ontologies. Table 3 shows for each of these ontologies the number of generating rules (G-rules), if it uses equality (Eq), expressivity (DL), and the number of classes (C), properties (P), and axioms (A).

### Materialisation Tests

To estimate the practicability of materialisation in acyclic ontologies, we measured the maximal depth of function symbol nesting in terms generated by skolem chase. This measure, which we call *ontology depth*, is of interest as it can be used to establish a bound on the size of the chase. Our tests revealed that most ontologies have small depths: out of the 124 MSA ontologies, 83 (66.9%) have depths less than 5; 13 (10.5%) have depths from 5 to 9; 24 (19.4%) have depths from 10 to 19; 2 (1.6%) have depths from 20 to 49; and 2 (1.6%) have depths from 50 to 80. This suggests that the materialisation of these ontologies might not be too large.

We also computed the materialisation of several acyclic ontologies. Since our implementation is only prototypical, our primary goal was not to evaluate the performance of materialisation, but rather to estimate the increase in ABox size. Although this increase may not be perfectly linear, we believe that it can be estimated by examining moderately-sized ABoxes. Most of our test ontologies, however, do not have substantial ABoxes; ontologies are often made available as general vocabularies, whereas ABoxes are application-specific and are thus usually not made publicly available. Because of that, we ran two kinds of experiments.

First, we computed the materialisation of two ontologies with nontrivial ABoxes: LUBM with one university and the 'kmi-basic-portal' ontology.[4] The TBox of LUBM contains 8 generating rules and has depth 1; the ABox before materialisation contains 100,543 facts. Materialisation required only 1 second, and it produced

[4] `http://kmi.open.ac.uk/semanticweb/ontologies/owl/kmi-basic-portal-ontology.owl`

| Depth | # | time | | gen. size | | mat. size | |
|---|---|---|---|---|---|---|---|
| | | max | avg | max | avg | max | avg |
| $< 5$ | 82 | 69 | 0.9 | 27 | 2 | 35 | 5 |
| 5–9 | 13 | 68 | 11 | 37 | 11 | 41 | 13 |
| 10–80 | 14 | 549 | 101 | 281 | 51 | 283 | 53 |

Table 4: Materialisation times (in seconds) and sizes

150,530 new facts (47,798 were added by generating rules). The 'kmi-basic-portal' ontology has 10 generating rules and has depth 2; the ABox contains 179 facts. Materialisation required only 0.01 seconds, and it added 975 new facts (with 151 added by generating rules).

Second, for each of the 124 ontologies identified as MSA we computed an ABox by instantiating each class and property with fresh individuals. We then computed the materialisation and measured the *generated size* (number of facts introduced by generating rules, divided by the facts in the initial ABox), the *materialisation size* (facts in the materialisation, divided by facts in the initial ABox), and the materialisation time. Since most generating rules in these ontologies had singleton body atoms (i.e., they are of the form $A(x) \rightarrow \exists R.C(x)$), these measures should provide a reasonable estimate of the increase in ABox size caused by materialisation. Of the 124 ontologies tested, 15 exceeded the $1,000s$ time limit for materialisation. The results for the other 109 ontologies are shown in Table 4. Ontologies are grouped by depth; each group shows the number of ontologies (#), and materialisation times, generated sizes, and materialisation sizes.

Thus, materialisation seems practically feasible for many ontologies: for the 82 ontologies with depth less than 5, materialisation increases the ontology size by a factor of 5. This suggests that principled, materialisation-based reasoning for ontologies beyond the OWL 2 RL profile may be feasible, especially for ontologies with relatively small depths.

## Conclusion

In this paper, we have studied the problem of CQ answering over acyclic existential rules. We have proposed two novel acyclicity conditions that are sufficient to ensure chase termination and which generalise all existing acyclicity conditions that we know of.

We have then studied the problem of CQ answering over acyclic DL ontologies. Acyclicity provides several compelling benefits for DL query answering. First, the CQ answering problem over Horn ontologies becomes computationally easier; second, under acyclicity conditions it is possible to extend Horn ontologies with arbitrary SWRL rules without affecting neither decidability nor worst-case complexity; finally, acyclicity enables principled extensions of ontology materialisation-based reasoners; furthermore, since many existing ontologies turn out to be acyclic, our results open the door for practical CQ answering beyond the OWL 2 RL profile.

## References


Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. F., eds. 2007. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd edition.

Baget, J.-F.; Leclère, M.; Mugnier, M.-L.; and Salvat, E. 2011. On rules with existential variables: Walking the decidability line. *Artificial Intelligence* 175(9–10):1620–1654.

Baget, J.-F.; Mugnier, M.-L.; and Thomazo, M. 2011. Towards farsighted dependencies for existential rules. In Rudolph, S., and Gutierrez, C., eds., *Proc. of the 5th Int. Conf. on Web Reasoning and Rule Systems (RR 2011)*, volume 6902 of *LNCS*, 30–45. Springer.

Beeri, C., and Vardi, M. Y. 1981. The implication problem for data dependencies. In *Proc. of the 8th Colloquium on Automata, Languages and Programming (ICALP 1981)*, 73–85.

Bishop, B., and Bojanov, S. 2011. Implementing OWL 2 RL and OWL 2 QL rule-sets for OWLIM. In Dumontier, M., and Courtot, M., eds., *Proc. of the OWL: Experiences and Directions Workshop (OWLED 2011)*, volume 796 of *CEUR WS Proceedings*.

Calì, A.; Gottlob, G.; Lukasiewicz, T.; Marnette, B.; and Pieris, A. 2010. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *Proc. of LICS*, 228–242.

Calì, A.; Gottlob, G.; and Pieris, A. 2010. Query answering under non-guarded rules in Datalog+/-. In Hitzler, P., and Lukasiewicz, T., eds., *Proc. of the 4th Int. Conf. on Web Reasoning and Rule Systems (RR 2010)*, volume 6333 of *LNCS*, 1–17. Springer.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. Automated Reasoning* 39(3):385–429.

Cuenca Grau, B.; Horrocks, I.; Motik, B.; Parsia, B.; Patel-Schneider, P.; and Sattler, U. 2008. OWL 2: The next step for OWL. *J. Web Semantics (JWS)* 6(4):309–322.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33(3):374–425.

Deutsch, A.; Nash, A.; and Remmel, J. B. 2008. The chase revisited. In *Proc. of the 27th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2008)*, 149–158.

Fagin, R.; Kolaitis, P. G.; Miller, R. J.; and Popa, L. 2005. Data exchange: semantics and query answering. *Theoretical Computer Science* 336(1):89–124.

Gardiner, T.; Tsarkov, D.; and Horrocks, I. 2006. Framework for an automated comparison of description logic reasoners. In Cruz, I. F.; Decker, S.; Allemang, D.; Preist, C.; Schwabe, D.; Mika, P.; Uschold, M.; and Aroyo, L., eds., *Proc. of the 5th Int. Semantic Web Conf. (ISWC 2006)*, volume 4273 of *LNCS*, 654–667. Springer.

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

Horrocks, I., and Patel-Schneider, P. F. 2004. A proposal for an OWL rules language. In *Proc. of the 13th Int. World Wide Web Conf. (WWW 2004)*, 723–731. ACM Press.

Hustadt, U.; Motik, B.; and Sattler, U. 2005. Data complexity of reasoning in very expressive description logics. In Kaelbling, L., and Saffiotti, A., eds., *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, 466–471. Professional Book Center.

Johnson, D. S., and Klug, A. C. 1984. Testing containment of conjunctive queries under functional and inclusion dependencies. *J. Comput. Syst. Sci.* 28(1):167–189.

Kiryakov, A.; Ognyanov, D.; and Manov, D. 2005. OWLIM: A pragmatic semantic repository for OWL. In Dean, M.; Guo, Y.; Jun, W.; Kaschek, R.; Krishnaswamy, S.; Pan, Z.; and Sheng, Q. Z., eds., *WISE Workshops*, 182–192.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2011. The combined approach to ontology-based data access. In Walsh, T., ed., *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI 2011)*, 2656–2661. AAAI Press/IJCAI.

Krötzsch, M., and Rudolph, S. 2011. Extending decidable existential rules by joining acyclicity and guardedness. In Walsh, T., ed., *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI 2011)*, 963–968. AAAI Press/IJCAI.

Krötzsch, M.; Rudolph, S.; and Hitzler, P. 2007a. Complexity boundaries for Horn description logics. In *Proc. of the 22nd National Conf. on Artificial Intelligence (AAAI 2007)*, 452–457. AAAI Press.

Krötzsch, M.; Rudolph, S.; and Hitzler, P. 2007b. Conjunctive queries for a tractable fragment of OWL 1.1. In Aberer, K.; Choi, K.-S.; Noy, N.; Allemang, D.; Lee, K.-I.; Nixon, L.; Golbeck, J.; Mika, P.; Maynard, D.; Mizoguchi, R.; Schreiber, G.; and Cudré-Mauroux, P., eds., *Proc. of the 6th Int. Semantic Web Conference. (ISWC 2007)*, volume 4825 of *LNCS*, 310–323. Springer.

Kutz, O.; Horrocks, I.; and Sattler, U. 2006. The Even More Irresistible $\mathcal{SROIQ}$. In Doherty, P.; Mylopoulos, J.; and Welty, C. A., eds., *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, 68–78. AAAI Press.

Lutz, C.; Toman, D.; and Wolter, F. 2009. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In Boutilier, C., ed., *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*. IJCAI.

Maier, D.; Mendelzon, A. O.; and Sagiv, Y. 1979. Test-

ing implications of data dependencies. *ACM Trans. Database Syst.* 4(4):455–469.

Marnette, B. 2009. Generalized schema-mappings: from termination to tractability. In *Proc. of the 28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2009)*, 13–22.

Meditskos, G., and Bassiliades, N. 2008. Combining a DL reasoner and a rule engine for improving entailment-based OWL reasoning. In Sheth, A.; Staab, S.; Dean, M.; Paolucci, M.; Maynard, D.; Finin, T.; and Thirunarayan, K., eds., *Proc. of the 7th Int. Semantic Web Conf. (ISWC 2008)*, volume 5318 of *LNCS*, 277–292. Springer.

Meier, M.; Schmidt, M.; and Lausen, G. 2009. On chase termination beyond stratification. *Proceedings of VLDB 2009* 2(1):970–981.

Motik, B.; Shearer, R.; and Horrocks, I. 2009. Hypertableau reasoning for description logics. *J. Artif. Intell. Res. (JAIR)* 36:165–228.

Ortiz, M.; Calvanese, D.; and Eiter, T. 2008. Data complexity of query answering in expressive description logics via tableaux. *J. Automated Reasoning* 41(1):61–98.

Ortiz, M.; Rudolph, S.; and Simkus, M. 2011. Query answering in the Horn fragments of the description logics $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$. In Walsh, T., ed., *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI 2011)*, 1039–1044.

Pérez-Urbina, H.; Motik, B.; and Horrocks, I. 2010. Tractable query answering and rewriting under description logic constraints. *J. Applied Logic* 8(2):186–209.

Rudolph, S., and Glimm, B. 2010. Nominals, inverses, counting, and conjunctive queries or: Why infinity is your friend! *J. Artif. Intell. Res. (JAIR)* 39:429–481.

Spezzano, F., and Greco, S. 2010. Chase termination: A constraints rewriting approach. *Proceedings of VLDB 2010* 3(1):93–104.

Wu, Z.; Eadon, G.; Das, S.; Chong, E. I.; Kolovski, V.; Annamalai, M.; and Srinivasan, J. 2008. Implementing an inference engine for RDFS/OWL constructs and user-defined rules in Oracle. In *Proc. of the 2008 IEEE 24th Int. Conf. on Data Engineering (ICDE 2008)*, 1239–1248. IEEE Computer Society.