Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Do Repeat Yourself: Understanding Sufficient Conditions for Restricted Chase Non-Termination

Lukas Gerlach[1] , **David Carral**[2]
[1]Knowledge-Based Systems Group, TU Dresden, Dresden, Germany
[2]LIRMM, Inria, University of Montpellier, CNRS, Montpellier, France
lukas.gerlach@tu-dresden.de, david.carral@inria.fr

Abstract

The disjunctive restricted chase is a sound and complete procedure for solving boolean conjunctive query entailment over knowledge bases of disjunctive existential rules. Alas, this procedure does not always terminate and checking if it does is undecidable. However, we can use acyclicity notions (sufficient conditions that imply termination) to effectively apply the chase in many real-world cases. To know if these conditions are as general as possible, we can use cyclicity notions (sufficient conditions that imply non-termination). In this paper, we discuss some issues with previously existing cyclicity notions, propose some novel notions for non-termination by dismantling the original idea, and empirically verify the generality of the new criteria.

1 Introduction

The *(disjunctive) chase* (Bourhis et al. 2016) is a sound and complete bottom-up materialization procedure to reason with *knowledge bases* (KBs) featuring *(disjunctive existential) rules*. In some cases, we can apply the chase to determine if a conjunctive query or a fact is a consequence of a KB under standard first-order semantics.

Example 1. *Consider the KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$ where $\mathcal{R}$ is the rule set $\{(1\text{–}4)\}$ and $\mathcal{D}$ is the database $\{\mathsf{Engine}(d)\}$.*

$$\mathsf{Engine}(x) \to \big(\exists v.\mathsf{IsIn}(x, v) \wedge \mathsf{Bike}(v)\big) \vee \mathsf{Spare}(x) \quad (1)$$

$$\mathsf{Bike}(x) \to \exists w.\mathsf{Has}(x, w) \wedge \mathsf{Engine}(w) \quad (2)$$

$$\mathsf{IsIn}(x, y) \to \mathsf{Has}(y, x) \quad (3)$$

$$\mathsf{Has}(x, y) \to \mathsf{IsIn}(y, x) \quad (4)$$

We can apply the chase procedure to verify if the fact $\mathsf{Spare}(c)$ *follows from $\mathcal{K}$. In this case, the* restricted chase *produces a universal model set with two models for $\mathcal{K}$; namely,* $\{\mathsf{Engine}(d), \mathsf{Spare}(d)\}$ *and* $\{\mathsf{Engine}(d), \mathsf{IsIn}(d, f_v(d)), \mathsf{Bike}(f_v(d)), \mathsf{Has}(f_v(d), d)\}$, *where $f_v(d)$) is a fresh term introduced to satisfy the existential quantifier in* (1)*. Since the second model does not contain* $\mathsf{Spare}(d)$*, this fact is not entailed by $\mathcal{K}$.*

Since boolean conjunctive query entailment is undecidable (Beeri and Vardi 1981), the chase may not always terminate. Even worse, we cannot decide if the chase terminates on a particular KB or if a rule set $\mathcal{R}$ is *acyclic* (Gogacz and Marcinkowski 2014; Grahne and Onet 2018); that is, if the chase terminates for every KB with $\mathcal{R}$. We can still verify rule set termination in practice using *acyclicity notions*; that is, sufficient conditions that imply termination (Fagin et al. 2005; Marnette 2009; Krötzsch and Rudolph 2011; Cuenca Grau et al. 2013; Carral, Dragoste, and Krötzsch 2017; Baget et al. 2014; Karimi, Zhang, and You 2021). However, if an acyclicity notion is not able to classify a rule set $\mathcal{R}$ as terminating, we never know if this notion is just not "general enough" or if the rule set is indeed non-terminating.

To address this issue, we study *cyclicity notions*, which imply non-termination. As a long-term motivation, these approaches can also help to fix potential modelling mistakes. To the best of our knowledge, only one such notion has been proposed for the restricted chase variant, namely *Restricted Model Faithful Cyclicity (RMFC)*. Alas, many non-terminating rule sets with disjunctions are not classified as such by this notion (Carral, Dragoste, and Krötzsch 2017). Worse still, the correctness proof of RMFC does not hold in its presented form (see Section 7). Recently, Gerlach and Carral have also proposed *Disjunctive Model Faithful Cyclicity (DMFC)* for the skolem chase. Note however that a cyclicity notion for the skolem chase is not directly a valid condition for restricted non-termination since the former variant terminates less often than the latter.

In this paper, our overarching goal is to improve our understanding of existing cyclicity notions such as RMFC and DMFC. We reconsider the underlying ideas and dismantle them into an extensible framework. We provide examples to explain how these notions work and clarify why certain technical details are necessary. As more tangible contributions, *(i)* we come up with novel cyclicity notions named *restricted prefix cyclicity (RPC)* and *deterministic RPC (DRPC)*, and *(ii)* we empirically evaluate the generality of these two.

To this aim, the key points of the sections are as follows:[1]

S3. *Cyclicity sequences* that guarantee *never-termination* by making use of *g-unblockability*.
S4. Checkable conditions that ensure g-unblockability.
S5. *Cyclicity prefixes* that allow cyclicity sequences.
S6. The notion (D)RPC that guarantees a cyclicity prefix.
S7. Detailed relations to DMFC and RMFC in particular.
S8. Empirical evaluation of the generality of (D)RPC.

[1]More proof details are online (Gerlach and Carral 2023b).

2 Preliminaries

We define Preds, Funs, Cons, and Vars to be mutually disjoint, countably infinite sets of predicates, function symbols, constants, and variables, respectively. Every $s \in \texttt{Preds} \cup \texttt{Funs}$ is associated with an *arity* $\mathsf{ar}(s) \geq 1$. The set $\texttt{Terms} \supseteq \texttt{Cons} \cup \texttt{Vars}$ contains $f(t_1, \ldots, t_n)$ for every $n \geq 1$, every $f \in \texttt{Funs}$ with $\mathsf{ar}(f) = n$, and every $t_1, \ldots, t_n \in \texttt{Terms}$. For some $\texttt{X} \in \{\texttt{Preds}, \texttt{Funs}, \texttt{Cons}, \texttt{Vars}, \texttt{Terms}\}$ and an expression ϕ, we write $\texttt{X}(\phi)$ to denote the set of all elements of X that syntactically occur in ϕ.

A term $t \notin \texttt{Vars} \cup \texttt{Cons}$ is *functional*. For a term t; let $\mathsf{depth}(t) = 1$ if t is not functional, and $\mathsf{depth}(t) = 1 + \max(\mathsf{depth}(s_1), \ldots, \mathsf{depth}(s_n))$ if t is of the form $f(s_1, \ldots, s_n)$. We write lists $t_1, \ldots, t_n$ of terms as $\boldsymbol{t}$ and often treat these as sets. A term s is a *subterm* of another term t if $t = s$, or t is of the form $f(\boldsymbol{s})$ and s is a subterm of some term in $\boldsymbol{s}$. For a term t, let $\mathsf{subterms}(t)$ be the set of all subterms of t. A term is *cyclic* if it has a subterm of the form $f(\boldsymbol{s})$ with $f \in \texttt{Funs}(\boldsymbol{s})$.

An *atom* is an expression of the form $\mathsf{P}(\boldsymbol{t})$ with P a predicate and $\boldsymbol{t}$ a list of terms such that $\mathsf{ar}(\mathsf{P}) = |\boldsymbol{t}|$. A *fact* is a variable-free atom. For a formula v, we write $v(\boldsymbol{x})$ to denote that $\boldsymbol{x}$ is the set of all free variables that occur in v.

Definition 1. *A* (disjunctive existential) rule *is a constant- and function-free first-order formula of the form*

$$\forall \boldsymbol{w}, \boldsymbol{x}.[\beta(\boldsymbol{w}, \boldsymbol{x}) \to \bigvee\nolimits_{i=1}^{n} \exists \boldsymbol{y}_i.\eta_i(\boldsymbol{x}_i, \boldsymbol{y}_i)] \qquad (5)$$

where $n \geq 1$; $\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}_1, \ldots,$ and $\boldsymbol{y}_n$ are pairwise disjoint lists of variables; $\bigcup_{i=1}^{n} \boldsymbol{x}_i = \boldsymbol{x}$; and $\beta, \eta_1, \ldots,$ and η_n are non-empty conjunctions of atoms.

We omit universal quantifiers when writing rules and often treat conjunctions as sets. The *frontier* of a rule ρ such as (5) is the variable set $\mathsf{frontier}(\rho) = \boldsymbol{x}$. Moreover, let $\mathsf{body}(\rho) = \beta$, let $\mathsf{head}_i(\rho) = \eta_i$ for every $1 \leq i \leq n$, and let $\mathsf{branching}(\rho) = n$. The rule ρ is *deterministic* if $n = 1$, *generating* if $\boldsymbol{y}_i$ is non-empty for some $1 \leq i \leq n$, and *datalog* if it is deterministic and non-generating.

A *(ground) substitution* is partial function from variables to ground terms; that is, to variable-free terms. We write $[x_1/t_1, \ldots, x_n/t_n]$ to denote the substitution mapping $x_1, \ldots, x_n$ to $t_1, \ldots, t_n$, respectively. For an expression ϕ and a substitution σ, let $\phi\sigma$ be the expression that results from ϕ by uniformly replacing every syntactic occurrence of every variable x by $\sigma(x)$ if the latter is defined.

For a rule ρ such as (5), let σ_ρ be the substitution mapping y to $f^{\rho}_{i,y}(\boldsymbol{x})$ for every $1 \leq i \leq n$ and every $y \in \boldsymbol{y}_i$ with $f^{\rho}_{i,y}$ a fresh function symbol unique for $\langle \rho, i, y \rangle$. If y uniquely identifies the tuple $\langle \rho, i, y \rangle$, we also write $f_y(\boldsymbol{x})$. (This is the case in all our examples.) The *skolemization* $\mathsf{sk}(\rho)$ of ρ is the expression $\beta \to (\bigvee_{i=1}^{n} \eta_i)\sigma_\rho$. For every $1 \leq i \leq n$, let $\mathsf{head}_i(\mathsf{sk}(\rho)) = \eta_i\sigma_\rho$.

A *trigger* λ is a pair $\langle \rho, \sigma \rangle$ with ρ a rule and σ a substitution with domain $\texttt{Vars}(\mathsf{body}(\rho))$. A trigger is *loaded* for a fact set $\mathcal{F}$ if $\mathsf{body}(\rho)\sigma \subseteq \mathcal{F}$. It is *obsolete* for $\mathcal{F}$ if there is a substitution τ that extends σ such that $\mathsf{head}_i(\rho)\tau \subseteq \mathcal{F}$ for some $1 \leq i \leq \mathsf{branching}(\rho)$. Let $\mathsf{out}_i(\lambda) = \mathsf{head}_i(\mathsf{sk}(\rho))\sigma$ for every $1 \leq i \leq \mathsf{branching}(\rho)$, and $\mathsf{out}(\lambda) = \{\mathsf{out}_i(\lambda) \mid 1 \leq i \leq \mathsf{branching}(\rho)\}$.

$$\begin{tikzcd}
\mathsf{Engine}(d) \arrow[r] \arrow[dr] & \ldots, \mathsf{Spare}(d)\langle(1), [x/d]\rangle \\
& \ldots, \mathsf{IsIn}(d, f_v(d)), \mathsf{Bike}(f_v(d))\langle(1), [x/d]\rangle \arrow[d] \\
& \ldots, \mathsf{Has}(f_v(d), d)\langle(3), [x/d, y/f_v(d)]\rangle
\end{tikzcd}$$

Figure 1: Chase Tree for Example 1

Consider a rule set $\mathcal{R}$. An *$\mathcal{R}$-term* is a term defined using the function symbols that occur in $\mathsf{sk}(\mathcal{R})$, some constants, and some variables. A substitution is an *$\mathcal{R}$-substitution* if its range is a set of $\mathcal{R}$-terms. An *$\mathcal{R}$-trigger* is a trigger with a rule from $\mathcal{R}$ and an $\mathcal{R}$-substitution.

A fact set $\mathcal{F}$ *satisfies* a rule ρ if all triggers with ρ are not loaded or obsolete for $\mathcal{F}$. A *knowledge base (KB)* is a pair $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$ of a rule set $\mathcal{R}$ and a *database* $\mathcal{D}$; that is, a function-free fact set. The *restricted chase* on input $\mathcal{K}$ exhaustively applies the outputs of triggers that are loaded and not obsolete in a tree with root $\mathcal{D}$ branching on disjunctions.

Restricted Chase We present a variant of the disjunctive chase (Bourhis et al. 2016) where the application of rules is *restricted*; that is, rules are only applicable if their heads are not obsolete with respect to previously derived facts. Moreover, we impose an order of rule applications that prioritises the application of (triggers with) datalog rules.

Definition 2. *A* chase tree *$T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ for a KB $\langle \mathcal{R}, \mathcal{D} \rangle$ is a directed tree where V is a set of vertices, E is a set of edges,* fct *is a function mapping vertices to fact sets, and* trg *is a function mapping every non-root vertex to a trigger. Moreover, the following hold:*

1. *For the root $r \in V$ of T, we have that $\mathsf{fct}(r) = \mathcal{D}$.*
2. *For every non-leaf vertex $v \in V$; there is an $\mathcal{R}$-trigger $\lambda = \langle \rho, \sigma \rangle$ that is loaded and not obsolete for $\mathsf{fct}(v)$ such that (i) the set $\mathsf{fct}(v)$ satisfies all datalog rules in $\mathcal{R}$ if the rule in λ is not datalog, (ii) v has exactly $n = \mathsf{branching}(\rho)$ children $c_1, \ldots, c_n$ (via E) with $\mathsf{fct}(c_i) = \mathsf{fct}(v) \cup \mathsf{out}_i(\lambda)$ and $\mathsf{trg}(c_i) = \lambda$ for each $1 \leq i \leq n$.*
3. *For every leaf vertex $v \in V$, the set $\mathsf{fct}(v)$ satisfies all of the rules in $\mathcal{R}$. For every $\mathcal{R}$-trigger λ that is loaded for $\mathsf{fct}(v)$ for some $v \in V$, there is a $k \geq 0$ such that λ is obsolete for $\mathsf{fct}(u)$ for each $u \in V$ reachable from v by a path of length k. That is, fairness.*

We refer to $\mathsf{fct}(v)$ and $\mathsf{trg}(v)$ for some $v \in V$ as the *fact-* and *trigger-label* of v, respectively. Informally, we say that a trigger (resp. a rule) is applied in a chase tree to signify that some vertex in the tree is labelled with this trigger (resp. a trigger with this rule).

Example 2. *The KB from Example 1 only admits one chase tree, which is depicted in Figure 1.*

A *branch* of a chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ is a maximal path in T starting at the root; that is, a vertex sequence $B = v_1, v_2, \ldots$ where v_1 is the root, $\langle v_i, v_{i+1} \rangle \in E$ for every $1 \leq i < |B|$, and the last element of B is a leaf if B is finite. The *result* of T is the set $\{\bigcup_{v \in B} \mathsf{fct}(v) \mid B \text{ a branch of } T\}$; chase results can be used to solve query entailment:

Proposition 1. *Consider the result $\mathfrak{R}$ of some (arbitrarily chosen) chase tree of a $\mathcal{K}$. Then, $\mathcal{K}$ entails a query $\gamma = \exists \boldsymbol{y}.\beta$ iff $\mathcal{F} \models \gamma$ for every $\mathcal{F} \in \mathfrak{R}$ iff for every $\mathcal{F} \in \mathfrak{R}$ there is a substitution σ with $\beta\sigma \subseteq \mathcal{F}$.*

Therefore, it is interesting to know if a rule set admits finite chase trees. A rule set $\mathcal{R}$ *terminates* if every chase tree of every KB with $\mathcal{R}$ is finite, it *sometimes-terminates* if every KB with $\mathcal{R}$ admits a finite chase tree, and it *never-terminates* if some KB with $\mathcal{R}$ has no finite chase trees.

We use skolem function names to backtrack the facts along which a term t appears in the chase; that is, the *birth facts* of t. For a constant c, let $\mathsf{BirthF}_{\mathcal{R}}(c) = \emptyset$; for a rule set $\mathcal{R}$ and an $\mathcal{R}$-term t of the form $f^{\rho}_{i,v}(\boldsymbol{s})$, let $\mathsf{BirthF}_{\mathcal{R}}(t) = \mathsf{out}_i(\langle \rho, \sigma \rangle) \cup \bigcup_{s \in \boldsymbol{s}} \mathsf{BirthF}_{\mathcal{R}}(s)$ where σ is a substitution with $\mathsf{frontier}(\rho)\sigma = \boldsymbol{s}$. For a trigger $\langle \psi, \tau \rangle$, let $\mathsf{BirthF}_{\mathcal{R}}(\langle \psi, \tau \rangle) = \bigcup_{x \in \mathsf{frontier}(\psi)} \mathsf{BirthF}_{\mathcal{R}}(\tau(x))$. The *term-skeleton* $\mathsf{skeleton}_{\mathcal{R}}(\langle \psi, \tau \rangle)$ of $\langle \psi, \tau \rangle$ consists of $\mathtt{Terms}(\mathsf{BirthF}_{\mathcal{R}}(\langle \psi, \tau \rangle))$ and every constant c with $c = \tau(x)$ for an $x \in \mathsf{frontier}(\psi)$.

Example 3. *Let $\mathcal{R} = \{(1), (2)\}$ and $\lambda = \langle (2), [x/f_v(d)] \rangle$. We have $\mathsf{BirthF}_{\mathcal{R}}(\lambda) = \{\mathsf{IsIn}(d, f_v(d)), \mathsf{Bike}(f_v(d))\}$ and $\mathsf{skeleton}_{\mathcal{R}}(\lambda) = \{d, f_v(d)\}$. (Note again that $f_v = f^{(1)}_{1,v}$.)*

3 Cyclicity Sequences

In this section, we introduce the notion of a *cyclicity sequence* Λ for a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$ (see Definition 6) and show that its existence implies that $\mathcal{K}$ only admits infinite chase trees (see Theorem 1). Intuitively, Λ is an infinite sequence of $\mathcal{R}$-triggers that are applied in some (infinite) branch of every chase tree of $\mathcal{K}$. To identify these branches, we consider the following notion from (Gerlach and Carral 2023c).

Definition 3. *A* head-choice *for a rule set $\mathcal{R}$ is a function* hc *that maps every rule $\rho \in \mathcal{R}$ to an element of $\{1, \ldots, \mathsf{branching}(\rho)\}$. For a rule $\rho \in \mathcal{R}$ and a trigger λ with ρ, we write $\mathsf{head}_{\mathsf{hc}}(\rho)$ and $\mathsf{out}_{\mathsf{hc}}(\lambda)$ instead of $\mathsf{head}_{\mathsf{hc}(\rho)}(\rho)$ and $\mathsf{out}_{\mathsf{hc}(\rho)}(\lambda)$, respectively. For a chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ of a KB with $\mathcal{R}$, we define $\mathsf{branch}(T, \mathsf{hc}) = v_1, v_2, \ldots$ as the branch of T such that $\mathsf{fct}(v_{i+1}) = \mathsf{fct}(v_i) \cup \mathsf{out}_{\mathsf{hc}}(\mathsf{trg}(v_{i+1}))$ for every $1 \leq i < |\mathsf{branch}(T, \mathsf{hc})|$; note that every branch starts with the root.*

Cyclicity sequences must satisfy three requirements (see Definition 6); let's start with the first two:

Definition 4. *Consider a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$, a head-choice* hc *for $\mathcal{R}$, and a sequence $\Lambda = \lambda_1, \lambda_2, \ldots$ of $\mathcal{R}$-triggers.*

- *Let $\mathcal{F}_0(\mathcal{K}, \mathsf{hc}, \Lambda) = \mathcal{D}$, and let $\mathcal{F}_{i+1}(\mathcal{K}, \mathsf{hc}, \Lambda) = \mathcal{F}_i(\mathcal{K}, \mathsf{hc}, \Lambda) \cup \mathsf{out}_{\mathsf{hc}}(\lambda_{i+1})$ for every $0 \leq i < |\Lambda|$.*
- *The sequence Λ is* loaded *for $\mathcal{K}$ and* hc *if λ_{i+1} is loaded for $\mathcal{F}_i(\mathcal{K}, \mathsf{hc}, \Lambda)$ for every $0 \leq i < |\Lambda|$.*
- *The sequence Λ is* growing *for $\mathcal{K}$ and* hc *if, for every $0 \leq i < |\Lambda|$, there is some $j > i$ and a term that occurs in $\mathcal{F}_j(\mathcal{K}, \mathsf{hc}, \Lambda)$ but not in $\mathcal{F}_i(\mathcal{K}, \mathsf{hc}, \Lambda)$. Note that Λ is infinite if this requirement is satisfied.*

For some variants of the chase, existence of a loaded and growing sequence of $\mathcal{R}$-triggers for a KB and a head-choice may suffice to witness non-termination. This is not the case for the restricted chase:

Example 4. *Consider the KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$ from Example 1, and the head-choice* hc *that maps every rule in $\mathcal{R}$ to 1. Moreover, consider the infinite sequence $\Lambda = \lambda_1, \lambda_2, \ldots$ of triggers such that:*

- *Let $t_0 = d$, let $t_i = f_v(t_{i-1})$ for every odd $i \geq 1$, and let $t_i = f_w(t_{i-1})$ for every even $i \geq 1$.*
- *For every $i \geq 1$; let $\lambda_i = \{(1), [x/t_{i-1}]\}$ if i is odd, and $\lambda_i = \{(2), [x/t_{i-1}]\}$ otherwise.*

The (infinite) sequence Λ is loaded and growing; however, the KB $\mathcal{K}$ terminates! Note that this KB only admits one chase tree, which is finite (see Example 2).

The issue in the previous example is that the second trigger of Λ cannot be applied in any chase tree of $\mathcal{K}$; this is because we must apply datalog triggers with (3) before we apply triggers with (2). To address this issue, we introduce a third requirement for cyclicity sequences (see Definition 6):

Definition 5. *Consider a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$, a head-choice* hc *for $\mathcal{R}$, and a sequence of $\mathcal{R}$-triggers $\Lambda = \lambda_1, \lambda_2, \ldots$*

- *An $\mathcal{R}$-trigger λ is* g-unblockable[2] *for $\mathcal{K}$ and* hc *if, for every chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ and every v in $\mathsf{branch}(T, \mathsf{hc})$ such that λ is loaded for $\mathsf{fct}(v)$, there is some u in $\mathsf{branch}(T, \mathsf{hc})$ with $\mathsf{out}_{\mathsf{hc}}(\lambda) \subseteq \mathsf{fct}(u)$. Intuitively, if λ is loaded for a vertex in the* hc*-branch of a chase tree T of $\mathcal{K}$, then its output according to* hc *eventually appears in this branch.*
- *If every trigger in Λ is g-unblockable, then this sequence is* g-unblockable *for $\mathcal{K}$ and* hc.

The sequence Λ in Example 4 is not g-unblockable because its second trigger does not satisfy this property. However, Λ is g-unblockable for a slightly different input KB:

Example 5. *Consider the rule set $\mathcal{R} = \{(1), (2)\}$; and the database $\mathcal{D}$, the head-choice* hc, *and the sequence Λ from Example 4. Since $\mathcal{R}$ contains neither (3) nor (4), the sequence Λ is g-unblockable for $\langle \mathcal{R}, \mathcal{D} \rangle$ and* hc. *Note that the KB $\langle \mathcal{R}, \mathcal{D} \rangle$ only admits one chase tree, which is infinite, and hence, $\mathcal{R}$ is never-terminating.*

We are ready to define cyclicity sequences:

Definition 6. *A sequence $\Lambda = \lambda_1, \lambda_2, \ldots$ of $\mathcal{R}$-triggers is a* cyclicity sequence *of a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$ and a head-choice* hc *if Λ is (infinite,) loaded, growing, and g-unblockable. Note that Λ is infinite if it is growing.*

Theorem 1. *A rule set $\mathcal{R}$ never-terminates if there is a cyclicity sequence for a KB with $\mathcal{R}$ and some head-choice.*

Proof. Assume that there is a cyclicity sequence $\Lambda = \lambda_1, \lambda_2, \ldots$ of some KB such as $\mathcal{K} = \langle \mathcal{R}, \mathcal{D} \rangle$ and some head-choice hc, and consider some chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ of $\langle \mathcal{R}, D \rangle$ and the sequence $\mathsf{branch}(T, \mathsf{hc}) = v_1, v_2, \ldots$ To show the theorem, we prove that the fact set $\mathcal{F}(T, \mathsf{hc}) = \bigcup_{i \geq 0} \mathsf{fct}(v_i)$ is infinite, which implies that both $\mathsf{branch}(T, \mathsf{hc})$ and T are infinite. This holds by infinity of $\bigcup_{i \geq 0} \mathcal{F}_i(\mathcal{K}, \mathsf{hc}, \Lambda)$ (since Λ is growing) and $\mathcal{F}_i(\mathcal{K}, \mathsf{hc}, \Lambda) \subseteq \mathcal{F}(T, \mathsf{hc})$ for every $i \geq 0$ (which follows by induction since Λ is loaded and g-unblockable.) $\square$

[2]The "g-" prefix stands for "general-"; we introduce more specific unblockability notions in the following section.

4 Infinite Unblockable Sequences

Theorem 1 provides a blueprint to show never-termination of a rule set $\mathcal{R}$: One "simply" has to show that $\mathcal{R}$ admits a cyclicity sequence (see Section 6). To show that such sequences exist, we have developed techniques to find infinite sequences of triggers that are g-unblockable. This is a rather challenging task; note that we cannot even decide if a single trigger is g-unblockable (by reduction from fact entailment (Beeri and Vardi 1981)):

Theorem 2. *The problem of deciding if a trigger is g-unblockable for a KB and a head-choice is undecidable.*

In this section, we first discuss ways to detect if a trigger is g-unblockable in some cases. Then, we devise strategies to show that some infinite sequences of triggers are g-unblockable, i.e. that unblockability "propagates".

Detecting Unblockability To detect if a trigger λ is g-unblockable, we make use of chase over-approximations before the application of λ:

Definition 7. *Consider a rule set $\mathcal{R}$, a head-choice* hc, *and some $\mathcal{R}$-trigger $\lambda = \langle \rho, \sigma \rangle$. A fact set $\mathcal{F}$ is an* over-approximation *of $\mathcal{R}$ and* hc *before λ if there is a function h over the set of terms such that (i) $h(\sigma(x)) = \sigma(x)$ for each $x \in \mathsf{frontier}(\rho)$ and, (ii) for every $u \in \mathsf{branch}(T, \mathsf{hc})$ in every chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ of every KB $\langle \mathcal{R}, \mathcal{D} \rangle$ with $\mathsf{out}_{\mathsf{hc}}(\lambda) \nsubseteq \mathsf{fct}(u)$, we have that $h(\mathsf{fct}(u)) \subseteq \mathcal{F}$.*

Intuitively, a chase over-approximation such as $\mathcal{F}$ above for $\mathcal{R}$ and hc before λ is some sort of "upper-bound" of all of the facts that can possibly occur in the label of a vertex in the hc-branch if this label does not include the output of λ. If λ is not obsolete for $\mathcal{F}$, its output eventually appears in the hc-branch of the chase:

Lemma 1. *If λ is not obsolete for some over-approximation of a rule set $\mathcal{R}$ and a head-choice* hc *before λ, then λ is g-unblockable for* hc *and every KB with $\mathcal{R}$.*

Proof. To show the contrapositive, we assume that $\lambda = \langle \rho, \sigma \rangle$ is not g-unblockable for some $\langle \mathcal{R}, \mathcal{D} \rangle$ and hc. Then, there exists a chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg} \rangle$ of $\langle \mathcal{R}, \mathcal{D} \rangle$ and hc such that *(i)* λ is loaded for some $v \in \mathsf{branch}(T, \mathsf{hc})$ and *(ii)* $\mathsf{out}_{\mathsf{hc}}(\lambda) \nsubseteq \mathsf{fct}(u)$ for every $u \in \mathsf{branch}(T, \mathsf{hc})$. Moreover, λ is obsolete for $\mathsf{fct}(w)$ for some $w \in \mathsf{branch}(T, \mathsf{hc})$ by Definition 2. By Definition 7, we find $h(\mathsf{fct}(w)) \subseteq \mathcal{F}$ for every over-approximation $\mathcal{F}$ of $\mathcal{R}$ and hc before λ with term mapping h. Then, $\langle \rho, h \circ \sigma \rangle$ is obsolete for $\mathcal{F}$; hence so is λ since $h \circ \sigma$ and σ agree on all frontier variables of ρ. ☐

Lemma 1 provides a strategy to detect g-unblockability for a given trigger λ: Compute some over-approximation $\mathcal{F}$ and then check if λ is obsolete for $\mathcal{F}$. We consider two alternative ways of computing these over-approximations:

Definition 8. *For a trigger λ, let $h^{\star}_{\lambda}$ be the function over the set of terms that maps every $t \in \mathsf{skeleton}_{\mathcal{R}}(\lambda)$ to itself and every other term to the special constant $\star$. Moreover, let h^{uc}_{λ} be another such function that maps every functional term $f(\boldsymbol{t}) \notin \mathsf{skeleton}_{\mathcal{R}}(\lambda)$ to a fresh constant c_f, every term in $\mathsf{skeleton}_{\mathcal{R}}(\lambda)$ and every constant of the form c_f to itself, and every other term to $\star$.*

For a rule set $\mathcal{R}$, a head-choice hc, *some $\mathcal{R}$-trigger $\lambda = \langle \rho, \sigma \rangle$, and some $h \in \{h^{\star}_{\lambda}, h^{uc}_{\lambda}\}$; let $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h)$ be the minimal fact set that*

1. *contains every fact that can be defined using a predicate occurring in $\mathcal{R}$ and constants in $\mathsf{skeleton}_{\mathcal{R}}(\lambda) \cup \{\star\}$,*
2. *includes $\mathsf{BirthF}_{\mathcal{R}}(\lambda)$, and*
3. *includes $h(\mathsf{out}_{\mathsf{hc}}(\lambda'))$ for every $\mathcal{R}$-trigger λ' such that λ' is loaded for $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h)$ and $\mathsf{out}_{\mathsf{hc}}(\lambda') \neq \mathsf{out}_{\mathsf{hc}}(\lambda)$.*

Moreover, we define $\mathcal{O}(\mathcal{R}, \lambda, h)$ as the minimal fact set that satisfies (1) and (2) above, and includes $h(\bigcup \mathsf{out}(\lambda'))$ for every $\mathcal{R}$-trigger $\lambda' = \langle \psi, \tau \rangle$ such that λ' is loaded for $\mathcal{O}(\mathcal{R}, \lambda, h)$ and if $\psi = \rho$, then $\mathsf{out}_i(\lambda') \neq \mathsf{out}_i(\lambda)$ for some $1 \leq i \leq \mathsf{branching}(\rho)$.

Intuitively, $\mathcal{O}(\mathcal{R}, \lambda, h)$ views outputs as if disjunctions were replaced by conjunctions in rules.

Example 6. *For the rule set $\mathcal{R} = \{(1), (2)\}$, the head-choice $\mathsf{hc}_1 : \mathcal{R} \to \{1\}$, some constant d, and the trigger $\lambda = \langle (2), [x/f_v(d)] \rangle$; the set $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_{\lambda})$ equals*

$$\{\mathsf{Engine}(s), \mathsf{Bike}(s), \mathsf{Spare}(s), \mathsf{IsIn}(s,t) \mid s,t \in \{\star, d\}\} \cup$$
$$\mathsf{BirthF}_{\mathcal{R}}(\lambda) \cup \{\mathsf{IsIn}(\star, c_v), \mathsf{Bike}(c_v), \mathsf{Has}(\star, c_w),$$
$$\mathsf{Has}(d, c_w), \mathsf{Engine}(c_w), \mathsf{IsIn}(c_w, c_v), \mathsf{Has}(c_v, c_w)\}.$$

In the above, we write c_v and c_w to refer to the fresh constants unique for f_v and f_w, respectively. Note that $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_{\lambda})$ does not contain $\mathsf{IsIn}(d, c_v)$ since h^{uc}_{λ} maps $f_v(d)$ to itself, or $\mathsf{Has}(f_v(d), c_w)$ since this fact is in the output of a trigger excluded by Item 3 in Definition 8. The set $\mathcal{O}(\mathcal{R}, \lambda, h^{uc}_{\lambda})$ includes $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_{\lambda})$ and additionally contains $\mathsf{Spare}(c_w)$. If we replace all occurrences of c_v and c_w in $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_{\lambda})$ (resp. $\mathcal{O}(\mathcal{R}, \lambda, h^{uc}_{\lambda})$) with $\star$, we obtain $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{\star}_{\lambda})$ (resp. $\mathcal{O}(\mathcal{R}, \lambda, h^{\star}_{\lambda})$).

Lemma 2. *For a rule set $\mathcal{R}$, a head-choice* hc, *an $\mathcal{R}$-trigger $\lambda = \langle \rho, \sigma \rangle$, and some $h \in \{h^{\star}_{\lambda}, h^{uc}_{\lambda}\}$; $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h)$ and $\mathcal{O}(\mathcal{R}, \lambda, h)$ are over-approximations of $\mathcal{R}$ and* hc *before λ.*

Proof. We show that $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h)$ is an over-approximation of $\mathcal{R}$ and hc before λ. The first condition of Definition 7 holds for h by Definition 8. For every $u \in \mathsf{branch}(T, \mathsf{hc})$ in every chase tree T of every KB $\langle \mathcal{R}, \mathcal{D} \rangle$, we can verify the second condition via induction on the path $u_1, \ldots, u_n$ in T with the root u_1 and $u_n = u$ assuming that $\mathsf{out}_{\mathsf{hc}}(\lambda) \nsubseteq \mathsf{fct}(u)$. The base case holds since the facts in $h(\mathcal{D})$ are contained in the facts defined by (1) in Definition 8. For the induction step it is important to realize that for each $2 \leq i \leq n$, $\mathsf{out}_{\mathsf{hc}}(\mathsf{trg}(u_i)) \neq \mathsf{out}_{\mathsf{hc}}(\lambda)$ holds by $\mathsf{out}_{\mathsf{hc}}(\lambda) \nsubseteq \mathsf{fct}(u)$.

For the second part of the claim, $\mathcal{O}(\mathcal{R}, \lambda, h)$ is an over-approximation of $\mathcal{R}$ and hc before λ since $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h) \subseteq \mathcal{O}(\mathcal{R}, \lambda, h)$ by Definition 8. ☐

Using the over-approximations from Definition 8, we define two different types of unblockability:

Definition 9. *Consider a rule set $\mathcal{R}$ and an $\mathcal{R}$-trigger λ. Then, λ is* $\star$-unblockable *for $\mathcal{R}$ if it features a datalog rule or it is not obsolete for $\mathcal{O}(\mathcal{R}, \lambda, h^{\star}_{\lambda})$. Moreover, it is* uc-unblockable *for $\mathcal{R}$ and some* hc *if it features a datalog rule or it is not obsolete for $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h^{uc}_{\lambda})$.*

Example 7. *Consider the rule set $\mathcal{R} = \{(1), (2)\}$ and the head-choice hc_1 mapping all rules to 1. The trigger $\lambda = \langle (2), [x/f_v(d)] \rangle$ is uc-unblockable for $\mathcal{R}$ and hc_1; it is not for $\mathcal{R}' = \mathcal{R} \cup \{(3)\}$ and hc_1. Note that $\mathcal{O}(\mathcal{R}', \mathsf{hc}_1, \lambda, h^{uc}_\lambda)$ includes $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_\lambda) \cup \{\mathsf{Has}(f_v(d), d)\}$ (among other facts). Therefore, λ is obsolete for $\mathcal{O}(\mathcal{R}', \mathsf{hc}_1, \lambda, h^{uc}_\lambda)$ but not for $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_\lambda)$.*

By Theorem 2, we cannot decide if a trigger λ is g-unblockable. However, we can effectively check $\star$- or *uc*-unblockability; both properties imply g-unblockability:

Lemma 3. *If an $\mathcal{R}$-trigger λ is $\star$-unblockable for a rule set $\mathcal{R}$, it is uc-unblockable for $\mathcal{R}$ and every head-choice. If λ is uc-unblockable for $\mathcal{R}$ and some head-choice hc, then it is g-unblockable for $\mathcal{R}$ and hc.*

Proof. The first implication holds since $\mathcal{O}(\mathcal{R}, \lambda, h^\star_\lambda)$ includes $h(\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h^{uc}_\lambda))$ with h the function that maps every fresh constant in the range of h^{uc}_λ to $\star$. For the second, note that a trigger λ with a datalog rule is g-unblockable. For the non-datalog case, we can apply Lemmas 1 and 2. $\square$

By Lemma 3, a trigger is *g*-unblockable if it is $\star$-unblockable; we can also prove this directly with Lemmas 1 and 2. By Lemma 3, *uc*-unblockability is more general than $\star$-unblockability; the other direction does not hold:

Example 8. *Consider the following rule set $\mathcal{R}$:*

$$R(x, y) \to \exists u.R(y, u) \tag{6}$$
$$R(x, y) \to \exists v.S(y, v) \tag{7}$$
$$R(x, y) \to \exists w.T(y, w) \tag{8}$$
$$S(x, y) \wedge T(x, y) \to R(x, y) \tag{9}$$

The trigger $\langle (6), [x/c_y, y/f_u(c_y)] \rangle$ is uc-unblockable but not $\star$-unblockable: We find $S(f_u(c_y), \star), T(f_u(c_y), \star)$ and therefore $R(f_u(c_y), \star)$ in $\mathcal{O}(\mathcal{R}, \lambda, h^\star_\lambda)$. On the other hand, we only find $S(f_u(c_y), c_v)$ and $T(f_u(c_y), c_w)$ but no fact of the form $R(f_u(c_y), \dots)$ in $\mathcal{O}(\mathcal{R}, \mathsf{hc}_1, \lambda, h^{uc}_\lambda)$.

Carral, Dragoste, and Krötzsch and Gerlach and Carral introduce similar notions to $\star$-unblockability in (2017) and (2023c). Here, we not only present a more general criterion (see Definition 9), but a blueprint to produce more comprehensive notions (see Definitions 7, 8 and Lemmas 1, 2).

Propagating Unblockability A key feature of *uc*/$\star$-unblockability is that these properties propagate across a *reversible* constant-mappings: (Gerlach and Carral 2023c)

Definition 10. *A constant mapping g is a partial function mapping constants to terms. For an expression ϕ, let $g(\phi)$ be the expression that results from replacing all syntactic occurrences of every constant c in the domain of g with $g(c)$.*

Consider a (possibly finite) set $\mathcal{T}$ of terms that contains every subterm of every $t \in \mathcal{T}$. A constant mapping g is reversible *for $\mathcal{T}$ if the following hold:*

1. *The function g is defined for every constant in $\mathcal{T}$.*
2. *For every $t, s \in \mathcal{T}$ with $t \neq s$, we have that $g(t) \neq g(s)$.*
3. *For every constant $c \in \mathcal{T}$, every subterm s of $g(c)$, and every functional term $u \in \mathcal{T}$; we have that $g(u) \neq s$.*

Lemma 4. *Consider a rule set $\mathcal{R}$, a head-choice hc, an $\mathcal{R}$-trigger $\lambda = \langle \rho, \sigma \rangle$, and a constant mapping g reversible for $\mathsf{skeleton}_\mathcal{R}(\lambda)$. If $\langle \rho, g \circ \sigma \rangle$ is an $\mathcal{R}$-trigger and $\langle \rho, \sigma \rangle$ is uc/$\star$-unblockable for $\mathcal{R}$ [and hc], then so is $\langle \rho, g \circ \sigma \rangle$.*

Proof. We define g^{-1} as follows: For a term t, let $g^{-1}(t) = s$ if there is a term s that occurs in $\mathsf{skeleton}_\mathcal{R}(\lambda)$ with $g(s) = t$, $g^{-1}(t) = t$ otherwise if t is a constant that does not occur in $\mathtt{Cons}(\mathsf{skeleton}_\mathcal{R}(\langle \rho, g \circ \sigma \rangle))$ (i.e. g^{-1} is the identity on fresh constants introduced by $h^{uc}_{\dots}$), and $g^{-1}(t) = \star$ otherwise. Note that g^{-1} is well-defined because g is reversible (cond. 2) for $\mathsf{skeleton}_\mathcal{R}(\lambda)$.

Consider the sets $\mathcal{F}'$ and $\mathcal{G}'$ of all facts that can be defined using any predicate in $\mathcal{R}$ and the constants in $\mathtt{Cons}(\mathsf{skeleton}_\mathcal{R}(\lambda)) \cup \{\star\}$ and $\mathtt{Cons}(\mathsf{skeleton}_\mathcal{R}(\langle \rho, g \circ \sigma \rangle)) \cup \{\star\}$, respectively. Moreover, consider the fact sets: $\mathcal{F} = \mathsf{BirthF}_\mathcal{R}(\lambda) \cup \mathcal{F}'$ and $\mathcal{G} = \mathsf{BirthF}_\mathcal{R}(\langle \rho, g \circ \sigma \rangle) \cup \mathcal{G}'$. Also, let the functions $\langle h_\mathcal{F}, h_\mathcal{G} \rangle$ be from $\{\langle h^\star_\lambda, h^\star_{\langle \rho, g \circ \sigma \rangle} \rangle, \langle h^{uc}_\lambda, h^{uc}_{\langle \rho, g \circ \sigma \rangle} \rangle\}$.

First claim: $g^{-1}(\mathcal{G}) \subseteq \mathcal{F}$. Since $g^{-1}(\mathcal{G}') \subseteq \mathcal{F}$ follows trivially, we only show $g^{-1}(\mathsf{BirthF}_\mathcal{R}(t)) \subseteq \mathcal{F}$ for every $t \in \mathtt{Terms}(\mathcal{G})$ via induction over the structure of terms. If t is a constant, then $g^{-1}(\mathsf{BirthF}_\mathcal{R}(t)) = \emptyset$; hence, the base case trivially holds. Regarding the induction step, consider a term t that is of the form $f^\psi_{\ell,y}(\boldsymbol{s})$:

Let $\boldsymbol{z}$ be the list of existentially quantified variables in $\mathsf{head}_\ell(\psi)$. Let τ be a substitution with $\mathsf{frontier}(\psi)\tau = \boldsymbol{s}$. Moreover, let $H = \mathsf{head}_\ell(\psi)\tau$. We only need to show that $g^{-1}(H) \subseteq \mathcal{F}$ to verify the induction step. We observe: If $g^{-1}(f^\psi_{\ell,z}(\boldsymbol{s}))$ is functional for some $z \in \boldsymbol{z}$, then $g^{-1}(f^\psi_{\ell,z'}(\boldsymbol{s})) = f^\psi_{\ell,z'}(g^{-1}(\boldsymbol{s}))$ for each $z' \in \boldsymbol{z}$ (†). Now, we perform a case by case analysis on $g^{-1}(t)$: If $g^{-1}(t)$ is a functional term, then $g^{-1}(H) = g^{-1}(\mathsf{head}_\ell(\psi)\tau) = \mathsf{head}_\ell(\psi)(g^{-1} \circ \tau) \subseteq \mathcal{F}$ (by †). If $g^{-1}(t) \in \mathtt{Cons} \setminus \{\star\}$, then $g^{-1}(f^\psi_{\ell,z}(\boldsymbol{s}))$ must be a constant for every $z \in \boldsymbol{z}$ (by †). Since g is reversible for $\mathsf{skeleton}_\mathcal{R}(\lambda)$ (cond. 3), $g^{-1}(s)$ is also a constant (possibly $\star$) for every $s \in \boldsymbol{s}$. Hence, $g^{-1}(H) \subseteq \mathcal{F}' \subseteq \mathcal{F}$. If $g^{-1}(t) = \star$ and $g^{-1}(t')$ is a constant (or $\star$) for every $t' \in \mathtt{Terms}(H)$, then $g^{-1}(H) \subseteq \mathcal{F}' \subseteq \mathcal{F}$. The remaining case of $g^{-1}(t) = \star$ and $g^{-1}(t') \notin \mathtt{Cons}$ for a $t' \in \mathtt{Terms}(H)$, contradicts reversibility of g (cond. 3) since then, there must be a constant c with $t' \in \mathsf{subterms}(g(c))$.

Second claim: The set $g^{-1}(\mathcal{O}(\mathcal{R}, [\mathsf{hc},] \langle \rho, g \circ \sigma \rangle, h_\mathcal{G}))$ is a subset of $\mathcal{O}(\mathcal{R}, [\mathsf{hc},] \lambda, h_\mathcal{F})$. There exists a finite list of triggers $\langle \psi_1, \tau_1 \rangle, \dots, \langle \psi_m, \tau_m \rangle$ that yields $\mathcal{O}(\mathcal{R}, [\mathsf{hc},] \langle \rho, g \circ \sigma \rangle, h_\mathcal{G})$ from $\mathcal{G}$ according to Definition 8. With the first claim as a base case, we can show via induction that the triggers $\langle \psi_1, g^{-1} \circ \tau_1 \rangle, \dots, \langle \psi_m, g^{-1} \circ \tau_m \rangle$ can be used in the construction of $\mathcal{O}(\mathcal{R}, [\mathsf{hc},] \lambda, h_\mathcal{F})$.

We conclude that $\langle \rho, g \circ \sigma \rangle$ is *uc*/$\star$-unblockable for $\mathcal{R}$ [and hc] as follows: Suppose for a contradiction that $\langle \rho, g \circ \sigma \rangle$ is not *uc*/$\star$-unblockable. Then, ρ is not datalog and $\langle \rho, g \circ \sigma \rangle$ is obsolete for $\mathcal{O}(\mathcal{R}, \langle \rho, g \circ \sigma \rangle, h^\star_{\langle \rho, g \circ \sigma \rangle})$ [resp. $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \langle \rho, g \circ \sigma \rangle, h^{uc}_{\langle \rho, g \circ \sigma \rangle})$]. By the second claim above, we obtain that λ is obsolete for $\mathcal{O}(\mathcal{R}, \lambda, h^\star_\lambda)$ [resp. $\mathcal{O}(\mathcal{R}, \mathsf{hc}, \lambda, h^{uc}_\lambda)$]. Hence, λ is not *uc*/$\star$-unblockable which contradicts the premise of the lemma. $\square$

Condition 2 in Definition 10 admits an "inverse" of g on term-level. Lemma 4 breaks without it:

Example 9. *Consider the rule set $\mathcal{R} = \{(10\text{–}13)\}$ and the head-choice hc_1 mapping all rules to 1.*

$$P(x,y) \to \exists u.R(x,u) \wedge S(y,u) \qquad (10)$$
$$R(x,y) \to \exists v.T(y,v) \qquad (11)$$
$$R(x,y) \wedge S(x,y) \to T(y,x) \qquad (12)$$
$$T(x,y) \to P(y,y) \qquad (13)$$

Consider the substitution $\sigma = [x/c_x, y/f_u(c_x, c_y)]$; the trigger $\langle(11), \sigma\rangle$, which is uc-unblockable for $\mathcal{R}$ and hc_1; and the constant mapping g that maps c_x and c_y to $f_v(f_u(c_x, c_y))$, which does not satisfy (2) *in Definition 10 for $\mathcal{T} = \mathsf{skeleton}_{\mathcal{R}}(\langle(11), \sigma\rangle)$. Indeed, $\langle(11), g \circ \sigma\rangle$ is not uc-unblockable! Intuitively, rule* (12) *"blocks" rule* (11) *if rule* (10) *is applied with a substitution that maps x and y to the same term, which is the case for $g \circ \sigma$.*

Lemma 4 also breaks without condition 3:

Example 10. *Consider the head-choice hc_1 mapping all rules to 1, and the rule set $\mathcal{R}$ containing the following:*

$$A(x) \to \exists u.P(x,u) \qquad B(x) \to \exists v.Q(x,v)$$
$$C(x) \to \exists w.S(x,w) \qquad Q(x,y) \to T(x)$$
$$P(x,y) \to T(y) \vee \exists z.R(x,y,z) \qquad (14)$$

The trigger $\langle(14), [x/c, y/f_u(d)]\rangle$ is uc-unblockable; the constant mapping g that maps c to $f_w(f_v(f_u(d)))$ and d to itself satisfies conditions (1) *and* (2) *in Definition 10 for $\mathcal{T} = \mathsf{skeleton}_{\mathcal{R}}(\langle(14), [x/c, y/f_u(d)]\rangle)$. Condition* (3) *is violated because $f_u(d)$ is a subterm of $g(c)$ and $g(f_u(d)) = f_u(d)$. Indeed, $\langle(14), g \circ [x/c, y/f_u(d)]\rangle$ is not uc-unblockable! Intuitively, this is because the birth facts feature $Q(f_u(d), f_v(f_u(d)))$, which yields $T(f_u(d))$, thus "blocking" the trigger $\langle(14), [x/f_w(f_v(f_u(d))), y/f_u(d)]\rangle$.*

5 Cyclicity Prefixes

Our high-level strategy to show that a rule set $\mathcal{R}$ never-terminates is to find a cyclicity sequence (see Definition 6 and Theorem 1), which is challenging because it is infinite by definition. Instead, we construct a *cyclicity prefix* for $\mathcal{R}$, which is finite and still yields a cyclicity sequence.

Intuitively, a cyclicity prefix is a (finite) list of *uc*-unblockable triggers that, if subsequently applied to a starting database $\mathcal{D}$, produce an isomorphic copy of $\mathcal{D}$ that contains at least one new term. We can then repeat the prefix to obtain a cyclicity sequence. To limit the number of starting databases, we only consider minimal databases for that some trigger with a generating rule in $\mathcal{R}$ is loaded.

Definition 11. *The* rule-database *of a rule ρ is the database $\mathcal{D}_\rho = \mathsf{body}(\rho)\sigma_{uc}$ where σ_{uc} is a substitution that maps every variable x to a fresh constant c_x unique for x.*

Assume that rule ρ can indeed be applied twice when starting on $\mathcal{D}_\rho$ and that the second application yields a cyclic term. If the triggers applied in between the first and last application of ρ are *uc*-unblockable, then this finite sequence of triggers is a cyclicity prefix, which can be extended into a cyclicity sequence by applying Lemma 4.

Definition 12. *A* cyclicity prefix *for a rule set $\mathcal{R}$, a head-choice hc, and a rule ρ is a (finite) list of $\mathcal{R}$-triggers $\Lambda = \langle\rho_0, \sigma_0\rangle, \ldots, \langle\rho_n, \sigma_n\rangle$ such that:*

- *Both $\rho_0 = \rho$ and $\sigma_0 = \sigma_{uc}$.*
- *The sequence Λ is loaded for $\langle\mathcal{R}, \mathcal{D}_\rho\rangle$ and hc.*
- *Each trigger $\langle\rho_i, \sigma_i\rangle$ with $1 \leq i \leq n$ is uc-unblockable and $\langle\rho_0, \sigma_0\rangle$ is g-unblockable.*
- *Both $\rho_n = \rho$ and $\mathsf{out}_{\mathsf{hc}}(\langle\rho_n, \sigma_n\rangle)$ features a ρ-cyclic term; that is, a term t that of the form $f(\boldsymbol{s})$ with $f \in \mathtt{Funs}(\mathsf{sk}(\rho))$ and $f \in \mathtt{Funs}(\boldsymbol{s})$,*
- *The constant mapping g_Λ with $g_\Lambda \circ \sigma_0 = \sigma_n$ is reversible for $\mathsf{skeleton}_{\mathcal{R}}(\langle\rho_i, g_\Lambda^j \circ \sigma_i\rangle)$ for every $1 \leq i \leq n$ and every $j \geq 0$. Note that g_Λ^0 is the identity function over constants, and $g_\Lambda^i = g_\Lambda \circ g_\Lambda^{i-1}$ for every $i \geq 1$.*

We can extend a cyclicity prefix such as Λ above into an infinite sequence of triggers that are defined via composition with the constant-mapping g_Λ; afterwards, we show that this extension is a cyclicity sequence.

Definition 13. *Given a* cyclicity prefix $\Lambda = \langle\rho_0, \sigma_0\rangle, \ldots, \langle\rho_n, \sigma_n\rangle$ *for a rule set $\mathcal{R}$, a head-choice, and a rule in $\mathcal{R}$; let Λ^∞ be the (infinite) sequence $\langle\rho_0, \sigma_0\rangle$, $\langle\rho_1, \sigma_1^1\rangle, \ldots, \langle\rho_n, \sigma_n^1\rangle, \langle\rho_1, \sigma_1^2\rangle, \ldots, \langle\rho_n, \sigma_n^2\rangle, \ldots$ of $\mathcal{R}$-triggers with $\sigma_i^j = g_\Lambda^{j-1} \circ \sigma_i$ for every $1 \leq i \leq n$ and $j \geq 1$.*

Example 11. *The finite trigger sequence $\langle(1), [x/c_x]\rangle$, $\langle(2), [x/f_v(c_x)]\rangle$, $\langle(1), [x/f_w(f_v(c_x))]\rangle$ is a cyclicity prefix for the rule set $\{(1), (2)\}$, head-choice hc_1, and* (1)*.*

Theorem 3. *If Λ is a cyclicity prefix for a rule set $\mathcal{R}$, a head-choice hc, and some $\rho \in \mathcal{R}$; then Λ^∞ is a cyclicity sequence for $\langle\mathcal{R}, \mathcal{D}_\rho\rangle$ and hc and hence, $\mathcal{R}$ never-terminates.*

Proof. Assume that there is a cyclicity prefix $\Lambda = \langle\rho_0, \sigma_0\rangle, \ldots, \langle\rho_n, \sigma_n\rangle$ for $\mathcal{R}$, ρ, and hc; and consider the constant-mapping g_Λ introduced in Definition 12. To prove Theorem 3, we show that Λ^∞ is a cyclicity sequence of the KB $\mathcal{K} = \langle\mathcal{R}, \mathcal{D}_\rho\rangle$ and hc. Namely, we argue that Λ^∞ is (a) loaded, (b) growing, and (c) g-unblockable.

(a): We show that the trigger $\langle\rho_i, \sigma_i^j\rangle$ is loaded in Λ^∞ for every $1 \leq i \leq n$ via induction on $j \geq 1$. The base case holds since Λ is loaded. The induction step from $j-1$ to j holds since $g_\Lambda(\mathcal{F}_{(j-1)\times n+i}(\mathcal{K}, \mathsf{hc}, \Lambda^\infty))$ is included in $\mathcal{F}_{j\times n+i}(\mathcal{K}, \mathsf{hc}, \Lambda^\infty)$.

(b): Since $\mathsf{out}_{\mathsf{hc}}(\langle\rho_n, \sigma_n\rangle)$ features a ρ-cyclic term, there is some $x \in \mathsf{frontier}(\rho)$ such that $\sigma_n(x) = g_\Lambda(\sigma_0(x))$ is functional. Note that $\sigma_0(x)$ is a constant c, $\mathsf{depth}(c) = 0$ and $\mathsf{depth}(g_\Lambda(c)) \geq 1$. Furthermore, $g_\Lambda(c)$ features c as a subterm. Hence, $g_\Lambda^k(c) > g_\Lambda^{k-1}(c)$ for every $k \geq 1$. But then, by construction of Λ^∞, $g_\Lambda^k(c)$ occurs in $\mathcal{F}_j(\mathcal{K}, \mathsf{hc}, \Lambda^\infty)$ for some $j \geq 0$. Thus, Λ must be growing.

(c): We already know by assumption that $\langle\rho_0, \sigma_0\rangle$ is g-unblockable. We can show via induction over $j \geq 1$ that $\langle\rho_i, \sigma_i^j\rangle$ is *uc/$\star$*-unblockable for every $1 \leq i \leq n$. For the base case with $j = 1$, the claim holds by assumption. For the induction step from j to $j+1$, the claim follows from Lemma 4 since g_λ is reversible for $\mathsf{skeleton}_{\mathcal{R}}(\langle\rho_i, \sigma_i^j\rangle)$. By Lemma 3, the sequence Λ^∞ is g-unblockable. $\square$

6 Novel Cyclicity Notions

Theorem 3 provides a blueprint to show non-termination of a rule set $\mathcal{R}$: One simply has to show that $\mathcal{R}$ admits a (finite) cyclicity prefix. In this section, we present two different ways to do so, namely RPC and DRPC, which we then use to define several cyclicity notions.

Restricted Prefix Cyclicity We introduce *restricted prefix-cyclicity* as the most general notion that we can define using our previous considerations:

Definition 14. *For a rule set $\mathcal{R}$, a head-choice* hc*, and a rule $\rho \in \mathcal{R}$; let $\mathsf{RPC}(\mathcal{R}, \mathsf{hc}, \rho)$ be the fact set that includes the database $\mathcal{D}_\rho$, the set $\mathsf{out}_{\mathsf{hc}}(\langle \rho, \sigma_{uc}\rangle)$, and $\mathsf{out}_{\mathsf{hc}}(\lambda)$ for every $\mathcal{R}$-trigger $\lambda = \langle \psi, \tau\rangle$ such that*

- *there are no cyclic terms in the range of τ,*
- *the trigger λ is loaded for $\mathsf{RPC}(\mathcal{R}, \mathsf{hc}, \rho)$,*
- *the trigger λ is uc-unblockable for $\mathcal{R}$ and* hc*, and*
- *the substitution σ is injective if $\psi = \rho$.*

A rule set $\mathcal{R}$ is restricted prefix-cyclic *(RPC) if there is some head-choice* hc*, some (generating rule) $\rho \in \mathcal{R}$, and some ρ-cyclic term that occurs in $\mathsf{RPC}(\mathcal{R}, \mathsf{hc}, \rho)$.*

The first restriction ensures that $\mathsf{RPC}(\mathcal{R}, \mathsf{hc}, \rho)$ is finite. The second and third are necessary by Definition 12. Note that $\langle \rho, \sigma_{uc}\rangle$ also needs to be g-unblockable but it is not *uc*-unblockable by definition. We show later that g-unblockability for $\langle \rho, \sigma_{uc}\rangle$ still follows if we can find a ρ-cyclic term. The fourth restriction ensures that we can find a reversible constant mapping for the cyclicity-prefix. Example 9 shows a rule set that is terminating but would be wrongly marked as RPC if we omit the fourth restriction.

Theorem 4. *If a rule set $\mathcal{R}$ is RPC, then it never-terminates.*

Proof. By Definition 14, there exists a finite trigger sequence $\Lambda = \langle \rho, \sigma_{uc}\rangle, \lambda_1 = \langle \rho_1, \sigma_1\rangle, \dots, \lambda_n = \langle \rho_n, \sigma_n\rangle$ that yields a (first) ρ-cyclic term in $\mathsf{RPC}(\mathcal{R}, \mathsf{hc}, \rho)$ with $\rho_n = \rho$ and no cyclic term in the image of every substitution σ_i for $1 \leq i \leq n$, and a constant mapping g_Λ for that we find $g_\Lambda \circ \sigma_{uc} = \sigma_n$. Furthermore, Λ is loaded for $\langle \mathcal{R}, \mathcal{D}_\rho\rangle$ and hc, and the triggers $\lambda_1, \dots, \lambda_n$ are *uc*-unblockable.

To prove that Λ is a cyclicity-prefix for $\mathcal{R}$, hc, and ρ, it only remains to show that (A) g_Λ is reversible for $\mathsf{skeleton}_{\mathcal{R}}(\langle \rho_i, g_\Lambda^j \circ \sigma_i\rangle)$ for each $1 \leq i \leq n$ and $j \geq 0$ and that (B) $\langle \rho, \sigma_{uc}\rangle$ is g-unblockable.

(A): Considering Definition 10, we show conditions (1), (2), and (3). Since $\mathtt{Cons}(\mathsf{skeleton}_{\mathcal{R}}(\langle \rho_i, g_\Lambda^j \circ \sigma_i\rangle))$ may only feature constants from $\mathcal{D}_\rho$, (1) holds. For (2) and (3), we make the following observations:

The substitutions of the triggers in Λ do not feature cyclic terms. Hence, for every constant c in $\mathcal{D}_\rho$, the term $g_\Lambda(c)$ does not feature nested function symbols from $\mathsf{sk}(\rho)$. [3] We show that, for any functional term t in $\mathsf{skeleton}_{\mathcal{R}}(\langle \rho_i, g_\Lambda^j \circ \sigma_i\rangle)$ (for any j and i), the term $g(t)$ features nested function symbols from $\mathsf{sk}(\rho)$: Every non-datalog trigger without functional terms in frontier positions is not *uc*-unblockable. Hence, t must have a subterm of the form $f(\boldsymbol{c})$ such that f occurs in $\mathsf{sk}(\rho)$ and $\boldsymbol{c} = \sigma_{uc}(\mathsf{frontier}(\rho))$. Also, for some $x \in \mathsf{frontier}(\rho_n)$, $g_\Lambda(\sigma_{uc}(x))$ is a functional term from $\mathsf{out}_{\mathsf{hc}}(\langle \rho, \sigma_{uc}\rangle)$. Thus, $f(g_\Lambda(\boldsymbol{c})) \in \mathtt{subterms}(g(t))$ features nested function symbols from $\mathsf{sk}(\rho)$.

By Definition 14, $g_\Lambda \circ \sigma_{uc}$ is injective and in turn g_Λ is injective on the constants in $\mathcal{D}_\rho$. We show (2) that $g_\Lambda(t) \neq g_\Lambda(s)$ for every t, s in $\mathsf{skeleton}_{\mathcal{R}}(\langle \rho_i, g_\Lambda^j \circ \sigma_i\rangle)$ with $t \neq s$: If t and s are constants, then $g_\Lambda(t) \neq g_\Lambda(s)$ since g_Λ is injective. If t is a constant and s is functional (or vice versa), then $g_\Lambda(s)$ features nested function symbols from $\mathsf{sk}(\rho)$ and $g_\Lambda(t)$ does not (or vice versa) by the above observations. If t and s are functional terms of the form $f(\boldsymbol{t})$ and $h(\boldsymbol{s})$, respectively, with $f \neq h$; then $g_\Lambda(t) \neq g_\Lambda(s)$ holds. If t and s are functional terms (of finite depth) of the form $f(t_1, \dots, t_n)$ and $f(s_1, \dots, s_n)$, respectively; then $t_i \neq s_i$ for some $1 \leq i \leq n$ since $t \neq s$ and we can recurse into one of the cases for t_i, s_i.

For (3), consider $c \in \mathtt{Cons}(\mathsf{skeleton}_{\mathcal{R}}(\langle \rho_i, g_\Lambda^j \circ \sigma_i\rangle)) \subseteq \mathtt{Cons}(\mathcal{D}_\rho)$ and some $s \in \mathtt{subterms}(g_\Lambda(c))$. If there was a functional term $u \in \mathsf{skeleton}_{\mathcal{R}}(\langle \rho_i, g_\Lambda^j \circ \sigma_i\rangle)$ with $g_\Lambda(u) = s$, we obtain a contradiction from the above observations, as $s \in \mathtt{subterms}(g_\Lambda(c))$ does not feature nested function symbols from $\mathsf{sk}(\rho)$ but $g_\Lambda(u) = s$ does. Thus, (A) holds.

(B): In the remainder of the proof, we show that $\langle \rho, \sigma_{uc}\rangle$ is g-unblockable for $\langle \mathcal{R}, \mathcal{D}_\rho\rangle$ and hc. Suppose for a contradiction that $\langle \rho, \sigma_{uc}\rangle$ is not g-unblockable. We obtain the contradiction by showing that $\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle$ is not *uc*-unblockable.

There exists a chase tree $T = \langle V, E, \mathsf{fct}, \mathsf{trg}\rangle$ for $\langle \mathcal{R}, \mathcal{D}_\rho\rangle$ and hc such that $\mathsf{out}_{\mathsf{hc}}(\langle \rho, \sigma_{uc}\rangle) \not\subseteq \mathsf{fct}(u)$ for each $u \in \mathsf{branch}(T, \mathsf{hc})$. Note that $\langle \rho, \sigma_{uc}\rangle$ is loaded for $\mathsf{fct}(v)$ for every $v \in \mathsf{branch}(T, \mathsf{hc})$ since it is loaded for $\mathcal{D}_\rho$. There must exists a (first) $w \in \mathsf{branch}(T, \mathsf{hc})$ such that $\langle \rho, \sigma_{uc}\rangle$ is obsolete for $\mathsf{fct}(w)$. Consider the path $v_0, \dots, v_m$ in T with v_0 the root and $v_m = w$. Let $\langle \psi_1, \tau_1\rangle, \dots, \langle \psi_m, \tau_m\rangle = \mathsf{trg}(v_1), \dots, \mathsf{trg}(v_m)$. We aim to show that $h(\mathsf{out}_{\mathsf{hc}}(\langle \psi_i, h \circ g_\Lambda \circ \tau_i\rangle)) \subseteq O$ for every $1 \leq i \leq m$ where $h = h^{uc}_{\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle}$ and $O = \mathcal{O}(\mathcal{R}, \mathsf{hc}, \langle \rho, g_\Lambda \circ \sigma_{uc}\rangle, h)$. Then, we find that $\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle$ is not *uc*-unblockable, i.e. the desired contradiction.

First, we find that $h(g_\Lambda(\mathsf{fct}(v_0)) = h(\mathsf{body}(\rho)(g_\Lambda \circ \sigma_{uc})) \subseteq O$ by making use of the triggers in Λ: We have that $h(\mathsf{out}_{\mathsf{hc}}(\langle \rho, \sigma_{uc}\rangle)) = \mathsf{out}_{\mathsf{hc}}(\langle \rho, \sigma_{uc}\rangle) \subseteq \mathsf{BirthF}_{\mathcal{R}}(\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle)$. Also, $h(\mathcal{D}_\rho)$ is contained in the set of all facts that can be defined using any predicate and constants from $\mathtt{Cons}(\mathsf{skeleton}_{\mathcal{R}}(\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle)) \cup \{\star\}$. Since Λ is loaded, we can now show that $h(\mathsf{out}_{\mathsf{hc}}(\langle \rho_i, h \circ \sigma_i\rangle)) \subseteq O$ for every $1 \leq i \leq n$. It is important to realize that $\mathsf{out}_{\mathsf{hc}}(\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle) \neq \mathsf{out}_{\mathsf{hc}}(\langle \rho_i, h \circ \sigma_i\rangle)$ by the fact that λ_n is the first trigger to yield a ρ-cyclic term.

Now, $h(\mathsf{out}_{\mathsf{hc}}(\langle \psi_i, h \circ g_\Lambda \circ \tau_i\rangle)) \subseteq O$ for $1 \leq i \leq m$ can be verified given that $\mathsf{out}_{\mathsf{hc}}(\langle \psi_i, h \circ g_\Lambda \circ \tau_i\rangle) \neq \mathsf{out}_{\mathsf{hc}}(\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle)$. The latter is the case since $\mathsf{out}_{\mathsf{hc}}(\langle \rho, \sigma_{uc}\rangle) \not\subseteq \mathsf{fct}(w)$. Therefore, $\langle \rho, g_\Lambda \circ \sigma_{uc}\rangle$ is not *uc*-unblockable; contradiction. □

[3]Consider the rule $\rho = A(x) \to \exists y, z.R(x, y) \wedge S(x, z)$. Then, $f_y(f_z(c))$ features nested function symbols from $\mathsf{sk}(\rho)$ but $f_w(f_y(d), f_z(c))$ does not (assuming w occurs in another rule).

In practice, it is infeasible to compute RPC membership because of the exponential number of head-choices. While this does not influence the complexity bounds (see Theorem 7), this exponential effort manifests often in practice. Instead, we check RPC$_s$ that considers far fewer head-choices but still explores a meaningful portion of the search space by using each head-disjunct of each rule at least once:

Definition 15. *For some $i \geq 1$, let hc_i be the head-choice such that, for every rule ρ: If $i \leq \mathsf{branching}(\rho)$, then $\mathsf{hc}_i(\rho) = i$. Otherwise, $\mathsf{hc}_i(\rho) = \mathsf{branching}(\rho)$. For a rule set $\mathcal{R}$, let $\mathsf{branching}(\mathcal{R})$ be the smallest number such that $\mathsf{branching}(\mathcal{R}) \geq \mathsf{branching}(\rho)$ for every $\rho \in \mathcal{R}$.*

A rule set $\mathcal{R}$ is RPC$_s$ if there is some $1 \leq i \leq \mathsf{branching}(\mathcal{R})$, some (generating) $\rho \in \mathcal{R}$, and some ρ-cyclic term that occurs in $\mathsf{RPC}(\mathcal{R}, \mathsf{hc}_i, \rho)$.

By Definitions 14 and 15, a rule set $\mathcal{R}$ is RPC if it is RPC$_s$. Therefore, the following follows from Theorem 4:

Theorem 5. *If a rule set is RPC$_s$, then it never-terminates.*

Deterministic Restricted Prefix Cyclicity We introduce *deterministic RPC* as a less general version of RPC; our goal here is to produce a notion that is similar to RMFC (Carral, Dragoste, and Krötzsch 2017) (see Section 7). It is therefore our baseline in the evaluation (see Section 8). For example, the rule set $\{(1), (2)\}$ is RPC but not DRPC.

Definition 16. *For a rule set $\mathcal{R}$ and a deterministic rule $\rho \in \mathcal{R}$, let $\mathsf{DRPC}(\mathcal{R}, \rho)$ be a fact set that includes the database $\mathcal{D}_\rho$, the set $\mathsf{out}_1(\langle \rho, \sigma_{uc} \rangle)$, and $\mathsf{out}_1(\lambda)$ for every deterministic $\mathcal{R}$-trigger $\lambda = \langle \psi, \tau \rangle$ such that*

- *there are no cyclic terms in the range of τ,*
- *the trigger λ is loaded for $\mathsf{DRPC}(\mathcal{R}, \rho)$,*
- *the trigger λ is $\star$-unblockable for $\mathcal{R}$, and*
- *the substitution σ is injective if $\psi = \rho$.*

A rule set $\mathcal{R}$ is deterministic restricted prefix-cyclic *(DRPC) if there is some deterministic (generating) rule $\rho \in \mathcal{R}$ and some ρ-cyclic term that occurs in $\mathsf{DRPC}(\mathcal{R}, \rho)$.*

Theorem 6. *If a rule set $\mathcal{R}$ is DRPC, it never-terminates.*

Proof. By Lemma 3, every $\star$-unblockable trigger is also *uc*-unblockable for every head-choice. Therefore, we find $\mathsf{DRPC}(\mathcal{R}, \rho) \subseteq \mathsf{RPC}(\mathcal{R}, \mathsf{hc}, \rho)$ for every deterministic rule $\rho \in \mathcal{R}$ and every head-choice hc. Hence, if $\mathcal{R}$ is DRPC, it is RPC (even RPC$_s$) and the claim follows by Theorem 4. □

Complexity The complexity of checking cyclicity is dominated by the double-exponential number of (non-cyclic) terms that may occur during the check. That is, checking RPC, RPC$_s$, or DRPC requires at most a double-exponential number of steps of which each is possible in double-exponential time. Hardness follows similarly to MFA (Cuenca Grau et al. 2013, Theorem 8): We extend Σ_3 to Σ_4 by adding a fresh atom $P_\psi(\boldsymbol{y})$ to the head of every $\psi \in \Sigma_3$ where $\boldsymbol{y}$ is the list of all body variables in ψ. By this, we make sure that unblockability does not interfere with the original proof idea. The set Ω is then defined as $\Sigma_4 \cup \{\rho = R(w, x) \wedge B(x) \rightarrow \exists y.R(x, y) \wedge A(y)\}$. We find that $\langle \Sigma_4, \{A(a)\} \rangle \models B(a)$ iff Ω is RPC, RPC$_s$, or DRPC.

Theorem 7. *Checking (D)RPC$_{(s)}$ is 2*EXPTIME*-complete.*

7 Related Work

Our main goal in this paper is to develop very general cyclicity notions for the disjunctive restricted chase. To the best of our knowledge, the only such existing notion is *restricted model faithful cyclicity* (RMFC), which was introduced by Carral, Dragoste, and Krötzsch in (2017). While trying to extend RMFC, we noticed that the proof of Theorem 11 in (2017), [4] which states that RMFC rule sets do not terminate, is incorrect; correctness of the theorem remains open.

Example 12. *Consider the rule set $\mathcal{R} = \{(15–20)\}$.*

$$\mathsf{Cl}_1(x) \wedge \mathsf{Cl}_2(y) \rightarrow \exists u.\mathsf{Red}(x, u) \wedge \mathsf{Red}(y, u) \tag{15}$$
$$\mathsf{Cl}_1(x) \wedge \mathsf{Red}(x, z) \rightarrow \exists v.\mathsf{Gr}(x, v) \wedge \mathsf{Blu}(z, v) \tag{16}$$
$$\mathsf{Red}(y, z) \wedge \mathsf{Blu}(z, w) \wedge \mathsf{Gr}(x, w) \rightarrow \mathsf{Gr}(y, y) \tag{17}$$
$$\mathsf{Red}(y, z) \wedge \mathsf{Blu}(z, w) \wedge \mathsf{Gr}(x, w) \rightarrow \mathsf{Blu}(z, y) \tag{18}$$
$$\mathsf{Red}(y, z) \wedge \mathsf{Blu}(z, w) \wedge \mathsf{Gr}(x, w) \rightarrow \mathsf{Cl}_1(y) \tag{19}$$
$$\mathsf{Cl}_2(y) \wedge \mathsf{Gr}(y, w) \rightarrow \mathsf{Cl}_2(w) \tag{20}$$

By Definition 11 in (2017), the rule set $\mathcal{R}$ is RMFC because the fact set $\mathcal{F}_{(15)}$ features a (15)-cyclic term. As per the proof of Theorem 11 in (2017), the chase of $\langle \mathcal{R}, \mathcal{I}_{(15)} \rangle$ should "contain infinitely many applications of (15)". This is not the case; in fact, the result of the only chase tree of $\langle \mathcal{R}, \mathcal{I}_{(15)} \rangle$ is the set $\{\mathcal{F}\}$ of fact sets where:

$$\begin{aligned}\mathcal{F} =& \{\mathsf{Cl}_1(c_x), \mathsf{Cl}_2(c_y), \mathsf{Red}(c_x, t), \mathsf{Red}(c_y, t), \mathsf{Gr}(c_x, s),\\ & \mathsf{Blu}(t, s), \mathsf{Gr}(c_x, c_x), \mathsf{Blu}(t, c_x), \mathsf{Gr}(c_y, c_y), \mathsf{Blu}(t, c_y)\}\end{aligned}$$

In the above, $t = f^{(15)}_{1,u}(c_x, c_y)$ and $s = f^{(16)}_{1,v}(c_x, t)$.

The problem stems from issues with Lemma 10 in (2017), which states that some triggers will eventually be applied if they are loaded for some vertex in the chase.

Example 13. *By Definition 10 in (2017), a trigger such as $\lambda = \langle (16), [x/c_y, z/f^{(15)}_{1,u}(c_x, c_y)] \rangle$ with $c_x, c_y \in \mathtt{Cons}$ is unblockable for the rule set $\mathcal{R} = \{(15–20)\}$. One can verify that Lemma 10 in (2017) does not hold for this trigger and the KB $\langle \mathcal{R}, \{\mathsf{Cl}_1(c_x), \mathsf{Cl}_2(c_y)\} \rangle$. To do so, simply note that this trigger is loaded for the fact set $\mathcal{F}$ defined at the end of Example 12; however, $\mathcal{F}$ does not include $\mathsf{out}_1(\lambda)$. Also, note that λ is not uc/$\star$-unblockable.*

We have sought to "repair" RMFC by introducing DRPC. We believe that both coincide for most real-world rule sets.

Another point of reference for us is our previous work (Gerlach and Carral 2023c), where we have introduced *Disjunctive Model Faithful Cyclicity (DMFC)* for the (disjunctive) skolem chase. We reuse many key ideas (also for the proofs) from this work, e.g. the main results for unblockability and reversiblity. A necessary but straightforward change is the definition of obsoleteness. While the idea of cyclicity sequences and prefixes was used in the proofs in spirit, a proper formalisation had not been presented. Furthermore, *uc*-unblockability was not considered for DMFC. Let us also stress that a cyclicity notion for the skolem chase is not a sufficient condition for restricted non-termination. There are (many) rule sets that terminate for the restricted chase but not for the skolem chase.

[4]Henceforth, we simply use (2017) as an abbreviation of (Carral, Dragoste, and Krötzsch 2017).

8 Evaluation

We have made available all evaluation materials online[5] including source code, rule sets, result files, and scripts used to obtain the counts. In our experiments, we make use of a well-known sufficient condition for restricted chase termination to obtain an upper bound for the cyclicity notions.

Definition 17. *A term is k-*cyclic *for some $k \geq 1$ if it features $k+1$ nested occurrences of the same function symbol. For instance, $f(f(a))$ is 1-cyclic but $g(f(a), f(b))$ is not.*

A rule set $\mathcal{R}$ is RMFA$_k$ for some $k \geq 1$ if there are no k-cyclic terms in the fact set $\mathsf{RMFA}(\mathcal{R})$*, which is introduced in Definition 7 of (Carral, Dragoste, and Krötzsch 2017).*

Theorem 8. *RMFA$_k$ rule sets (with $k \geq 1$) terminate.*

The above result follows from the proof of Theorem 7 in (Carral, Dragoste, and Krötzsch 2017).

Test Suite We consider rule sets from the evaluation of (Gerlach and Carral 2023c), which were obtained from OWL ontologies via normalization and translation; see Section 6 in (Cuenca Grau et al. 2013) for more details. OWL axioms with "at-most restrictions" and "nominals" are dropped because their translation requires the use of equality. The ontologies come from the Oxford Ontology Repository (OXFD)[6] and the Manchester OWL Corpus (MOWL) (Matentzoglu et al. 2014). We ignore rule sets without generating rules since these are trivially terminating.

Results For every rule set $\mathcal{R}$ in our test suite; we checked if $\mathcal{R}$ is RMFA$_2$, DRPC, and RPC$_s$ using our implementations; we ran each check with a 4h timeout on a cloud instance with 8 threads and 32GB of RAM (comparable to a modern laptop). We present our results in Table 1. We split the rule sets that are purely deterministic ($\wedge$) from the ones containing disjunctions ($\vee$). We further split by the number of generating rules $\#\exists$ and present the total number of rule sets $\#$ for each bucket. For example, in the third row of the table, we indicate that there are 27 deterministic rule sets in OXFD with at least 20 and at most 99 generating rules of which 23 are RMFA$_2$, 2 are DRPC, and 3 are RPC$_s$ When conidering RMFA$_2$ together with DRPC or RPC$_s$, the percentages of rule sets that cannot be characterised as either terminating or non-terminating drop from DRPC to RPC$_s$. For MOWL $\wedge$, OXFD $\vee$, and MOWL $\vee$, the drops are from 5% to 3%, 37% to 6%, and 45% to 5%, respectively; for OXFD $\wedge$ the percentage is around 21% for both. Our improvements are significant on the datasets with disjunctions; for these, RPC is considerably more general than DRPC, (which we introduced as a replacement for RMFC).

While many of the non-classified rule sets simply result from timeouts, there are 38 rule sets in OXFD for which both RMFA$_2$ and DRPC finished without capturing the rule set. Analogously, with RPC$_s$, we find 7 rule sets. For MOWL the numbers are 1505 and 110. This indicates that there is still room for improvement on the theoretical side but also that timeouts are indeed a big issue, which happens often when we consider large datasets.

[5]https://doi.org/10.5281/zenodo.8005904 Gerlach and Carral

[6]https://www.cs.ox.ac.uk/isg/ontologies/

	#∃	# tot.	RMFA$_2$	DRPC	RPC$_s$
OXFD ∧	1–19	58	58	0	0+0
	20–99	27	23	2	2+1
	100–999	109	61	8	8+1
	1–999	**194**	**142**	**10**	**10+2**
MOWL ∧	1–19	1139	866	239	239+12
	20–99	269	228	27	27+5
	100-999	363	271	46	46+21
	1–999	**1771**	**1365**	**312**	**312+38**
OXFD ∨	1–19	37	32	0	0+5
	20–99	18	4	7	7+7
	100–999	147	8	13	13+20
	1–999	**102**	**44**	**20**	**20+32**
MOWL ∨	1–19	1361	806	48	48+405
	20–99	894	196	171	171+496
	100-999	1150	500	136	136+470
	1–999	**3405**	**1502**	**355**	**355+1371**

Table 1: Restricted Chase Termination: Generating Rule Sets

9 Conclusions and Future Work

We make three tangible contributions: *(i)* We define RPC; a very general cyclicity notion tailored for rule sets with disjunctions. *(ii)* We discovered problems with RMFC and defined DRPC as a "repaired" version of this notion. *(iii)* We present an evaluation to demonstrate the usefulness of our work. Beyond these three, we believe that our efforts provide a framework for interesting future work.

Extending Cyclicity Notions Despite the fact that RPC is more general than existing criteria, there are many rule sets in our evaluation that remain open; that is, rule sets cannot be characterised as terminating or non-terminating.

Our work provides three different main strategies to achieve possible extensions. The first one is to produce "weaker" over-approximations and thus a more general strategy to detect g-unblockability; even g-unblockability itself can be relaxed. The second is to generalize cyclicity prefixes; perhaps by checking loadedness from slightly different databases. For instance, the rule set $\mathcal{R}$ from Example 12 is neither DRPC nor RPC (as intended) but actually it is never-terminating; note that $\langle \mathcal{R}, \{\mathsf{Cl}_1(c), \mathsf{Cl}_2(c)\}\rangle$ does not admit finite chase trees. Thus, even correctness of RMFC remains an open problem. The third one is to develop more comprehensive search strategies to find cyclicity prefixes; for instance, we can relax the condition in the first item of Definition 14 to look a bit further.

Explaining Cyclicity In many real-world use-cases, the existence of infinite universal models highlights a modelling mistake. We can use RPC and DRPC as methods to explain the loss of termination. For instance, a cyclicity prefix as defined in Section 5 constitutes a small and clear explanation of one way of loosing termination. In the future, we aim to automatically compute minimal sets of rules that can be removed (or added!) to deactivate a cyclicity prefix.

Acknowledgements

Lukas is / has been funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) in project 389792660 (TRR 248, Center for Perspicuous Systems), by the Bundesministerium für Bildung und Forschung (BMBF, Federal Ministry of Education and Research) under European ITEA project 01IS21084 (InnoSale, Innovating Sales and Planning of Complex Industrial Products Exploiting Artificial Intelligence), by BMBF and DAAD (German Academic Exchange Service) in project 57616814 (SECAI, School of Embedded and Composite AI), and by the Center for Advancing Electronics Dresden (cfaed).

David is funded by the ANR project CQFD (ANR-18-CE23-0003).

References

Baget, J.; Garreau, F.; Mugnier, M.; and Rocher, S. 2014. Extending acyclicity notions for existential rules. In Schaub, T.; Friedrich, G.; and O'Sullivan, B., eds., *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014), Czech Republic*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, 39–44. IOS Press.

Beeri, C., and Vardi, M. Y. 1981. The implication problem for data dependencies. In Even, S., and Kariv, O., eds., *Proceedings of the 8th International Colloquium on Automata, Languages and Programming (ICALP 1981), Israel*, volume 115 of *Lecture Notes in Computer Science*, 73–85. Springer.

Bourhis, P.; Manna, M.; Morak, M.; and Pieris, A. 2016. Guarded-based disjunctive tuple-generating dependencies. *ACM Transactions On Database Systems (TODS)* 41(4):27:1–27:45.

Carral, D.; Dragoste, I.; and Krötzsch, M. 2017. Restricted chase (non)termination for existential rules with disjunctions. In Sierra, C., ed., *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017), Australia*, 922–928. ijcai.org.

Cuenca Grau, B.; Horrocks, I.; Krötzsch, M.; Kupke, C.; Magka, D.; Motik, B.; and Wang, Z. 2013. Acyclicity notions for existential rules and their application to query answering in ontologies. *Journal of Artificial Intelligence Resesearch (JAIR)* 47:741–808.

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.

Gerlach, L., and Carral, D. 2023a. Do repeat yourself: Understanding sufficient conditions for restricted chase non-termination: Evaluation materials. https://doi.org/10.5281/zenodo.8005904.

Gerlach, L., and Carral, D. 2023b. Do repeat yourself: Understanding sufficient conditions for restricted chase non-termination (technical report). https://iccl.inf.tu-dresden.de/web/Inproceedings3353.

Gerlach, L., and Carral, D. 2023c. General acyclicity and cyclicity notions for the disjunctive skolem chase. In Williams, B.; Chen, Y.; and Neville, J., eds., *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence, 2023, USA*, 6372–6379. AAAI Press.

Gogacz, T., and Marcinkowski, J. 2014. All-instances termination of chase is undecidable. In Esparza, J.; Fraigniaud, P.; Husfeldt, T.; and Koutsoupias, E., eds., *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP 2014), Denmark, Part II*, volume 8573 of *Lecture Notes in Computer Science*, 293–304. Springer.

Grahne, G., and Onet, A. 2018. Anatomy of the chase. *Fundamenta Informaticae* 157(3):221–270.

Karimi, A.; Zhang, H.; and You, J. 2021. Restricted chase termination for existential rules: A hierarchical approach and experimentation. *Theory and Practice of Logic Programming* 21(1):4–50.

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

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

Matentzoglu, N.; Tang, D.; Parsia, B.; and Sattler, U. 2014. The manchester OWL repository: System description. In Horridge, M.; Rospocher, M.; and van Ossenbruggen, J., eds., *Proceedings of the 13th International Semantic Web Conference (ISWC 2014), Posters & Demonstrations Track, Italy*, volume 1272 of *CEUR Workshop Proceedings*, 285–288. CEUR-WS.org.