Reasoning over Existential Rules with Acyclicity Notions and the Datalog-first Restricted Chase

David Carral

Slides available at https://iccl.inf.tu-dresden.de/web/Existential-rules-acyclicity
Preliminaries
Existential Rules

\[ \forall x, y, z . \left( \text{HasParent}(x, y) \land \text{HasSister}(y, z) \rightarrow \text{HasAunt}(x, z) \right) \]

\[ \forall x . \left( \text{Human}(x) \rightarrow \exists y . \left( \text{HasParent}(x, y) \land \text{Human}(y) \right) \right) \]

\[ \forall x, y, w . \left( \text{P}(x, a, y) \land \text{R}(y, w) \land \text{S}(w, x) \rightarrow \exists v . \left( \text{R}(w, v) \land \text{A}(v) \right) \right) \]
Existential Rules

\[
\text{HasParent}(x, y) \land \text{HasSister}(y, z) \rightarrow \text{HasAunt}(x, z)
\]

\[
\text{Human}(x) \rightarrow \exists y . \text{HasParent}(x, y) \land \text{Human}(y)
\]

\[
\text{P}(x, a, y) \land \text{R}(y, w) \land \text{S}(w, x) \rightarrow \exists v . \text{R}(w, v) \land \text{A}(v)
\]
Existential Rules

HasParent($x, y$) $\land$ HasSister($y, z$) $\rightarrow$ HasAunt($x, z$)

Human($x$) $\rightarrow$ $\exists y . \text{HasParent}(x, y) \land \text{Human}(y)$

$P(x, a, y) \land R(y, w) \land S(w, x)$ $\rightarrow$ $\exists v . R(w, v) \land A(v)$

Facts

HasFriend(stan, kyle)

$P(a, c, d)$
Existential Rules

\[
\text{HasParent}(x, y) \land \text{HasSister}(y, z) \rightarrow \text{HasAunt}(x, z)
\]
\[
\text{Human}(x) \rightarrow \exists y . \text{HasParent}(x, y) \land \text{Human}(y)
\]
\[
P(x, a, y) \land R(y, w) \land S(w, x) \rightarrow \exists v . R(w, v) \land A(v)
\]

Facts

- HasFriend(stan, kyle)
- P(a, c, d)

BCQs

- \exists x, y . \text{HasConflictOfInterest}(x, y)
- \exists x, y, z, w . P(x, y, z) \land R(x, w) \land A(w)
The Chase Algorithm

- \text{Features}(x, y) \rightarrow \text{Actor}(y)
- \text{ActsIn}(x, y) \rightarrow \text{Features}(y, x)
- \text{DirectedBy}(x, y) \rightarrow \text{Directs}(y, x)
- \text{Directs}(x, y) \land \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z)

\text{Director}(\text{spielberg})

\text{ActsIn}(\text{judeLaw}, \text{ai})

\text{DirectedBy}(\text{ai, spielberg})
The Chase Algorithm

\[ \text{Features}(x, y) \rightarrow \text{Actor}(y) \]
\[ \text{ActsIn}(x, y) \rightarrow \text{Features}(y, x) \]
\[ \text{DirectedBy}(x, y) \rightarrow \text{Directs}(y, x) \]
\[ \text{Directs}(x, y) \land \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z) \]

\[ \text{judeLaw} \]
\[ \text{actsIn(judeLaw, ai)} \]
\[ \text{directedBy(ai, spielberg)} \]
\[ \text{spielberg} \]
\[ \text{Director(spielberg)} \]
The Chase Algorithm

Features(x, y) → Actor(y)
ActsIn(x, y) → Features(y, x)

DirectedBy(x, y) → Directs(y, x)
Directs(x, y) ∧ Features(y, z) → DirectsActor(x, z)

DirectedBy(spielberg, judeLaw)
ActsIn(judeLaw, ai)

DirectedBy(ai, spielberg)
The Chase Algorithm

\[
\begin{align*}
\text{Features}(x, y) &\rightarrow \text{Actor}(y) \\
\text{ActsIn}(x, y) &\rightarrow \text{Features}(y, x) \\
\text{DirectedBy}(x, y) &\rightarrow \text{Directs}(y, x) \\
\text{Directs}(x, y) \land \text{Features}(y, z) &\rightarrow \text{DirectsActor}(x, z)
\end{align*}
\]
The Chase Algorithm

**Features**\( (x, y) \rightarrow \text{Actor}(y) \)

**ActsIn**\( (x, y) \rightarrow \text{Features}(y, x) \)

**DirectedBy**\( (x, y) \rightarrow \text{Directs}(y, x) \)

**Directs**\( (x, y) \wedge \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z) \)

The diagram illustrates the relationships between actors and directors, with arrows indicating direction and relationships such as "ActsIn" and "DirectedBy."
The Chase Algorithm

\[
\text{Features}(x, y) \rightarrow \text{Actor}(y) \\
\text{ActsIn}(x, y) \rightarrow \text{Features}(y, x) \\
\text{DirectedBy}(x, y) \rightarrow \text{Directs}(y, x) \\
\text{Directs}(x, y) \land \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z)
\]
The Chase Algorithm

- \( \text{Features}(x, y) \rightarrow \text{Actor}(y) \)
- \( \text{ActsIn}(x, y) \rightarrow \text{Features}(y, x) \)
- \( \text{DirectedBy}(x, y) \rightarrow \text{Directs}(y, x) \)
- \( \text{Directs}(x, y) \land \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z) \)
The Chase Algorithm

- \( \text{Features}(x, y) \rightarrow \text{Actor}(y) \)
- \( \text{ActsIn}(x, y) \rightarrow \text{Features}(y, x) \)
- \( \text{DirectedBy}(x, y) \rightarrow \text{Directs}(y, x) \)
- \( \text{Directs}(x, y) \land \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z) \)

**Diagram:**

- Jude Law: Actor
- Spielberg: Director
- ai
- DirectedBy
- ActsIn
- Features

Reasoning over Existential Rules with Acyclicity Notions

David Carral

4/33
The Chase Algorithm

Features(x, y) → Actor(y) 
ActsIn(x, y) → Features(y, x) 
DirectedBy(x, y) → Directs(y, x) 
Directs(x, y) ∧ Features(y, z) → DirectsActor(x, z)
The Chase Algorithm

\[ \text{Features}(x, y) \rightarrow \text{Actor}(y) \]
\[ \text{ActsIn}(x, y) \rightarrow \text{Features}(y, x) \]
\[ \text{DirectedBy}(x, y) \rightarrow \text{Directs}(y, x) \]
\[ \text{Directs}(x, y) \land \text{Features}(y, z) \rightarrow \text{DirectsActor}(x, z) \]
The Skolem Chase

\[
\begin{align*}
\text{Bicycle}(x) & \longrightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \longrightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]

b : Bicycle
The Skolem Chase

Bicycle(x) $\rightarrow$ HasPart(x, f_v(x)) $\land$ Wheel(f_v(x))
Wheel(x) $\rightarrow$ IsPartOf(x, f_w(x)) $\land$ Bicycle(f_w(x))

HasPart(x, y) $\rightarrow$ IsPartOf(y, x)
IsPartOf(x, y) $\rightarrow$ HasPart(y, x)

b : Bicycle
•
The Skolem Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) & \text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{Wheel}(x) & \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x)) & \text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

b : Bicycle

●
The Skolem Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \\
\text{Wheel}(x) & \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x))
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

\[b : \text{Bicycle}\]
\[v(b) : \text{Wheel}\]
The Skolem Chase

\[ \text{Bicycle}(x) \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \]
\[ \text{Wheel}(x) \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x)) \]
\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]

\[ b : \text{Bicycle} \]
\[ v(b) : \text{Wheel} \]
\[ w(v(b)) : \text{Bicycle} \]
The Skolem Chase

\[ \text{Bicycle}(x) \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \]
\[ \text{Wheel}(x) \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x)) \]

\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]

\[ b : \text{Bicycle} \]
\[ v(b) : \text{Wheel} \]
\[ w(v(b)) : \text{Bicycle} \]
The Skolem Chase

\[
\text{Bicycle}(x) \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x))
\]

\[
\text{Wheel}(x) \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x))
\]

\[
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]

\[
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]

\[
b : \text{Bicycle}
\]

\[
v(b) : \text{Wheel}
\]

\[
w(v(b)) : \text{Bicycle}
\]
The Skolem Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \\
\text{Wheel}(x) & \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x)) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Skolem Chase

Bicycle(x) → HasPart(x, v(x)) ∧ Wheel(v(x))
Wheel(x) → IsPartOf(x, w(x)) ∧ Bicycle(w(x))

HasPart(x, y) → IsPartOf(y, x)
IsPartOf(x, y) → HasPart(y, x)

\[ b : \text{Bicycle} \]
\[ v(b) : \text{Wheel} \]
\[ w(v(b)) : \text{Bicycle} \]
\[ v(w(v(b))) : \text{Wheel} \]
The Skolem Chase

\[ \text{Bicycle}(x) \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \]
\[ \text{Wheel}(x) \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x)) \]

HasPart(x, y) → IsPartOf(y, x)

IsPartOf(x, y) → HasPart(y, x)
The Skolem Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \text{HasPart}(x, \text{v}(x)) \land \text{Wheel}(\text{v}(x)) & \text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{Wheel}(x) & \rightarrow \text{IsPartOf}(x, \text{w}(x)) \land \text{Bicycle}(\text{w}(x)) & \text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

\[
\begin{align*}
b : \text{Bicycle} & \\
\text{HasPart} & \\
\text{IsPartOf} & \\
\text{v}(b) : \text{Wheel} & \\
\text{w}((\text{v}(b))) : \text{Bicycle} & \\
\text{IsPartOf} & \\
\text{v}(\text{w}((\text{v}(b)))) : \text{Wheel}
\end{align*}
\]
The Skolem Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \\
\text{Wheel}(x) & \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x))
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Skolem Chase

\[ \text{Bicycle}(x) \rightarrow \text{HasPart}(x, v(x)) \land \text{Wheel}(v(x)) \]
\[ \text{Wheel}(x) \rightarrow \text{IsPartOf}(x, w(x)) \land \text{Bicycle}(w(x)) \]

\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]

The diagram illustrates the relationships between the concepts of 'Bicycle', 'Wheel', 'HasPart' and 'IsPartOf'.
The Restricted Chase

Bicycle(x) → ∃v . HasPart(x, v) ∧ Wheel(v)
Wheel(x) → ∃w . IsPartOf(x, w) ∧ Bicycle(w)
HasPart(x, y) → IsPartOf(y, x)
IsPartOf(x, y) → HasPart(y, x)
The Restricted Chase

\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}

\text{b : Bicycle}
The Restricted Chase

\[
\text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v)
\]

\[
\text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]

\[
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]

\[
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
The Restricted Chase

\[ \text{Bicycle}(x) \longrightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \]
\[ \text{Wheel}(x) \longrightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]

\[ \text{HasPart}(x, y) \longrightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \longrightarrow \text{HasPart}(y, x) \]
The Restricted Chase

\[ \text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \]

\[ \text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]

\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]

\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]

\[
\begin{align*}
b : \text{Bicycle} & \quad \text{Bicycle} : w(v(b)) \\
\text{HasPart} & \quad \text{HasPart} \\
\text{IsPartOf} & \quad \text{IsPartOf} \\
v(b) : \text{Wheel} &
\end{align*}
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

\[ \text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \]
\[ \text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]
\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]
The Restricted Chase

\[
\text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v)
\]
\[
\text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]
\[
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]
\[
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]
The Restricted Chase

\[
\text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v)
\]
\[
\text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]
\[
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]
\[
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]

\[
b : \text{Bicycle}
\]

\[
\text{Bicycle} : w(v(b))
\]

\[
v(b) : \text{Wheel}
\]

\[
\text{b} : \text{Bicycle}
\]

\[
\text{2}
\]

\[
\text{1}
\]

\[
\text{2}
\]

\[
\text{1}
\]
The Restricted Chase

\[ \text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \]
\[ \text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]
\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v \cdot \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w \cdot \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Restricted Chase

- \( \text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \)
- \( \text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \)

- \( \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \)
- \( \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \)

\[ b : \text{Bicycle} \]

\[ \text{Bicycle} : w(v(b)) \]

\[ v(b) : \text{Wheel} \]

\[ b : \text{Bicycle} \]

\[ 2 \]

\[ 1 \]

\[ 2 \]

\[ 1 \]

\[ 4 \]

\[ 3 \]

\[ 4 \]

\[ 3 \]
Reasoning over Existential Rules with Acyclicity Notions

The Restricted Chase

\[ \text{Bicycle}(x) \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \]

\[ \text{Wheel}(x) \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]

\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]

\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]

\[ b : \text{Bicycle} \]

\[ \text{v}(b) : \text{Wheel} \]

\[ \text{Bicycle} : w(v(b)) \]

\[ b : \text{Bicycle} \]

\[ 1 \quad 2 \quad 3 \quad 4 \quad 2 \quad 1 \quad 2 \quad 1 \quad 3 \]

David Carral

6/33
The Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The Datalog-First Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

\[
\begin{align*}
b : \text{Bicycle} \\
\bullet
\end{align*}
\]
The Datalog-First Restricted Chase

Bicycle(x) → ∃v . HasPart(x, v) ∧ Wheel(v)
Wheel(x) → ∃w . IsPartOf(x, w) ∧ Bicycle(w)

HasPart(x, y) → IsPartOf(y, x)
IsPartOf(x, y) → HasPart(y, x)
The Datalog-First Restricted Chase

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
Acyclicity Notions
Acyclicity Notions

Restricted Chase (Non)Termination for Existential Rules with Disjunctions
David Carral, Irina Dragoste, and Markus Krötzsch
[IJCAI 2017]
Acyclicity Notions for Universal Termination

* Joint Acyclicity (JA) [IJCAI 2011]
* Model-Summarising Acyclicity (MSA) and Model-Faithful Acyclicity (MFA) [J. Artif. Intell. Res. 2013]
Acyclicity Notions for Universal Termination

* Joint Acyclicity (JA) [IJCAI 2011]
* Model-Summarising Acyclicity (MSA) and Model-Faithful Acyclicity (MFA) [J. Artif. Intell. Res. 2013]
Acyclicity Notions for Universal Termination


* Joint Acyclicity (JA) [IJCAI 2011]

* Model-Summarising Acyclicity (MSA) and Model-Faithful Acyclicity (MFA) [J. Artif. Intell. Res. 2013]

* Restricted Joint Acyclicity (RJA), Restricted Model-Summarising Acyclicity (RMSA), and Restricted Model-Faithful Acyclicity (RMFA) [IJCAI 2017]
Acylicity Notions for Universal Termination

* Joint Acyclicity (JA) [IJCAI 2011]
* Model-Summarising Acyclicity (MSA) and Model-Faithful Acyclicity (MFA) [J. Artif. Intell. Res. 2013]
* Restricted Joint Acyclicity (RJA), Restricted Model-Summarising Acyclicity (RMSA), and Restricted Model-Faithful Acyclicity (RMFA) [IJCAI 2017]
Acyclicity Notions for Universal Termination


* Joint Acyclicity (JA) [IJCAI 2011]

* Model-Summarising Acyclicity (MSA) and Model-Faithful Acyclicity (MFA) [J. Artif. Intell. Res. 2013]

* Restricted Joint Acyclicity (RJA), Restricted Model-Summarising Acyclicity (RMSA), and Restricted Model-Faithful Acyclicity (RMFA) [IJCAI 2017]
Acyclicity Notions for Universal Termination

* Joint Acyclicity (JA) [IJCAI 2011]
* Model-Summarising Acyclicity (MSA) and Model-Faithful Acyclicity (MFA) [J. Artif. Intell. Res. 2013]
* Restricted Joint Acyclicity (RJA), Restricted Model-Summarising Acyclicity (RMSA), and Restricted Model-Faithful Acyclicity (RMFA) [IJCAI 2017]

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists v . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists w . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check
The MFA Check

* Fact: If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.
The MFA Check

* Fact: If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* MFA Check: Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y \cdot \text{HasPart}(x, y) & \text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{Wheel}(x) & \rightarrow \exists y \cdot \text{IsPartOf}(x, w) \land \text{Bicycle}(w) & \text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, y) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \longrightarrow \exists y . \text{HasPart}(x, y) \\
\text{Wheel}(x) & \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* Fact: If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.
* MFA Check: Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) &\longrightarrow \exists y . \text{HasPart}(x, v) \\
\text{Wheel}(x) &\longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) &\longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) &\longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) &\rightarrow \exists y . \text{HasPart}(x, v) \\
\text{Wheel}(x) &\rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) &\rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) &\rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* Fact: If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* MFA Check: Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) &\quad \longrightarrow \exists y . \text{HasPart}(x, v) \\
\text{Wheel}(x) &\quad \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) &\quad \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) &\quad \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a **cyclic skolem term** (with a repeated function symbol) appears.

\[
\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \\
\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a **cyclic skolem term** (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y \ . \ \text{HasPart}(x, v) \\
\text{Wheel}(x) & \rightarrow \exists y \ . \ \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
**The MFA Check**

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a *cyclic skolem term* (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) &\rightarrow \exists y . \text{HasPart}(x, v) \\
\text{Wheel}(x) &\rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* Fact: If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant "★"), then it terminates on all sets of facts.

* MFA Check: Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \wedge \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \wedge \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
**The MFA Check**

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \longrightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y \ . \ \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y \ . \ \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.
* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[ \text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \]
\[ \text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]
\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]
The MFA Check

* **Fact**: If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check**: Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \longrightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

Bicycle(x) → ∃y . HasPart(x, v) ∧ Wheel(v)
Wheel(x) → ∃y . IsPartOf(x, w) ∧ Bicycle(w)

HasPart(x, y) → IsPartOf(y, x)
IsPartOf(x, y) → HasPart(y, x)
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

\[
\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a cyclic skolem term (with a repeated function symbol) appears.

---

The MFA Check

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

HasPart(x, y) \rightarrow IsPartOf(y, x)

IsPartOf(x, y) \rightarrow HasPart(y, x)
The MFA Check

* **Fact:** If the Skolem chase terminates on the critical instance (the set of all possible facts containing a single constant “★”), then it terminates on all sets of facts.

* **MFA Check:** Perform chase with the critical instance, check if it stops; give up if a **cyclic skolem term** (with a repeated function symbol) appears.

Bicycle(x) → ∃y . HasPart(x, v) ∧ Wheel(v)
Wheel(x) → ∃y . IsPartOf(x, w) ∧ Bicycle(w)
HasPart(x, y) → IsPartOf(y, x)
IsPartOf(x, y) → HasPart(y, x)

Not MFA!
The RMFA Check: Blocked Checks
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!

\[
\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v)
\]
\[
\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!

\[ \text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \]
\[ \text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \]
\[ \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \]
\[ \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x) \]
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the must have been derived to derive this fact. By checking these facts we can in some cases determine that the application of the rule and substitution that generates this fact is blocked.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the
  must have been derived to derive this fact. By checking these facts we can in some cases
determine that the application of the rule and substitution that generates this fact is **blocked**.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) & \text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) & \text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

**Example:** Suppose for a contradiction that the fact \(\text{Wheel}(v(w(t)))\) with \(t\) some term is
derived during the computation of a chase sequence.
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the
must have been derived to derive this fact. By checking these facts we can in some cases
determine that the application of the rule and substitution that generates this fact is blocked.

\[
\begin{align*}
Bicycle(x) & \rightarrow \exists y \cdot HasPart(x, v) \land Wheel(v) & HasPart(x, y) & \rightarrow IsPartOf(y, x) \\
Wheel(x) & \rightarrow \exists y \cdot IsPartOf(x, w) \land Bicycle(w) & IsPartOf(x, y) & \rightarrow HasPart(y, x)
\end{align*}
\]

**Example:** Suppose for a contradiction that the fact \(Wheel(v(w(t)))\) with \(t\) some term is
derived during the computation of a chase sequence.
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the must have been derived to derive this fact. By checking these facts we can in some cases determine that the application of the rule and substitution that generates this fact is **blocked**.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

**Example:** Suppose for a contradiction that the fact \(\text{Wheel}(v(w(t)))\) with \(t\) some term is derived during the computation of a chase sequence.
* Such a fact may only be derived via application of the **red rule** on \(\text{Bicycle}(w(t))\) which in turn may only be derived if the **blue rule** is applied. Hence, \(\text{Wheel}(t)\) and \(\text{IsPartOf}(t, w(t))\) and are also part of the chase before \(\text{Wheel}(v(w(t)))\) is derived.
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the must have been derived to derive this fact. By checking these facts we can in some cases determine that the application of the rule and substitution that generates this fact is blocked.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

Example: Suppose for a contradiction that the fact \(\text{Wheel}(v(w(t)))\) with \(t\) some term is derived during the computation of a chase sequence.
* Such a fact may only be derived via application of the red rule on \(\text{Bicycle}(w(t))\) which in turn may only be derived if the blue rule is applied. Hence, \(\text{Wheel}(t)\) and \(\text{IsPartOf}(t, w(t))\) and are also part of the chase before \(\text{Wheel}(v(w(t)))\) is derived.
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the must have been derived to derive this fact. By checking these facts we can in some cases determine that the application of the rule and substitution that generates this fact is blocked.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y \ . \ \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y \ . \ \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]

Example: Suppose for a contradiction that the fact \(\text{Wheel}(v(w(t)))\) with \(t\) some term is derived during the computation of a chase sequence.
* Such a fact may only be derived via application of the red rule on \(\text{Bicycle}(w(t))\) which in turn may only be derived if the blue rule is applied. Hence, \(\text{Wheel}(t)\) and \(\text{IsPartOf}(t, w(t))\) and are also part of the chase before \(\text{Wheel}(v(w(t)))\) is derived.
* Because the green rule is Datalog, \(\text{DirectedBy}(v(t), t)\) is also part of the chase.
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the must have been derived to derive this fact. By checking these facts we can in some cases determine that the application of the rule and substitution that generates this fact is **blocked**.

\[
\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \quad \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]
\[
\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \quad \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]

**Example:** Suppose for a contradiction that the fact \(\text{Wheel}(v(w(t)))\) with \(t\) some term is derived during the computation of a chase sequence.
* Such a fact may only be derived via application of the **red rule** on \(\text{Bicycle}(w(t))\) which in turn may only be derived if the **blue rule** is applied. Hence, \(\text{Wheel}(t)\) and \(\text{IsPartOf}(t, w(t))\) and are also part of the chase before \(\text{Wheel}(v(w(t)))\) is derived.
* Because the **green rule** is Datalog, \(\text{DirectedBy}(v(t), t)\) is also part of the chase.
The RMFA Check: Blocked Checks

* Problem: Datalog-first restricted chase termination is not monotone!
* In particular, it always terminates on the critical instance.
* Idea: for each fact that occurs in the chase sequence, we can re-trace a necessary fact set the must have been derived to derive this fact. By checking these facts we can in some cases determine that the application of the rule and substitution that generates this fact is blocked.

\[
\begin{align*}
\text{Bicycle}(x) \rightarrow & \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) \rightarrow & \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) \rightarrow & \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) \rightarrow & \text{HasPart}(y, x)
\end{align*}
\]

Example: Suppose for a contradiction that the fact $\text{Wheel}(v(w(t)))$ with $t$ some term is derived during the computation of a chase sequence.

* Such a fact may only be derived via application of the red rule on $\text{Bicycle}(w(t))$ which in turn may only be derived if the blue rule is applied. Hence, $\text{Wheel}(t)$ and $\text{IsPartOf}(t, w(t))$ and are also part of the chase before $\text{Wheel}(v(w(t)))$ is derived.
* Because the green rule is Datalog, $\text{DirectedBy}(v(t), t)$ is also part of the chase.
* The red rule may not be applied to introduce $\text{Director}(v(w(t)))$ since its application with respect to the substitution $\{x / w(t)\}$ is restricted.
The RMFA Check

\[
\begin{align*}
\text{Bicycle}(x) &\rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) &\rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) &\rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) &\rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.

\[
\begin{align*}
\text{Bicycle}(x) & \longrightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\text{Bicycle}(x) \rightarrow \exists y \cdot \text{HasPart}(x, v) \land \text{Wheel}(v)
\]
\[
\text{Wheel}(x) \rightarrow \exists y \cdot \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]
\[
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]
\[
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
Perform a chase-like construction on the critical instance.

Only apply an existential rule with respect to a substitution if this pair is not blocked.

Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) \rightarrow & \exists y \cdot \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) \rightarrow & \exists y \cdot \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) \rightarrow & \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) \rightarrow & \text{HasPart}(y, x)
\end{align*}
\]
**The RMFA Check**

- Perform a chase-like construction on the critical instance.
- Only apply an existential rule with respect to a substitution if this pair is not **blocked**.
- Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y \ . \ \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y \ . \ \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \\
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

HasPart(x, y) $\rightarrow$ IsPartOf(y, x)
IsPartOf(x, y) $\rightarrow$ HasPart(y, x)
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \quad \text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)
\]
\[
\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w) \quad \text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not **blocked**.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v)\]
\[\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)\]
\[\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x)\]
\[\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\text{Bicycle}(x) \longrightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v)
\]

\[
\text{Wheel}(x) \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\text{Bicycle}(x) \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\]

\[
\text{HasPart}(x, y) \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) \rightarrow \text{HasPart}(y, x)
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) & \longrightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \longrightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \longrightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \longrightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

* Perform a chase like construction on the critical instance.
* Only apply an existential rule with respect to a substitution if this pair is not blocked.
* Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) & \rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) & \rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

\[
\begin{align*}
\text{HasPart}(x, y) & \rightarrow \text{IsPartOf}(y, x) \\
\text{IsPartOf}(x, y) & \rightarrow \text{HasPart}(y, x)
\end{align*}
\]
The RMFA Check

- Perform a chase like construction on the critical instance.
- Only apply an existential rule with respect to a substitution if this pair is not blocked.
- Give up if the procedure does not stop before the occurrence of a cyclic term.

\[
\begin{align*}
\text{Bicycle}(x) &\rightarrow \exists y . \text{HasPart}(x, v) \land \text{Wheel}(v) \\
\text{Wheel}(x) &\rightarrow \exists y . \text{IsPartOf}(x, w) \land \text{Bicycle}(w)
\end{align*}
\]

HasPart\(x, y\) $\rightarrow$ IsPartOf\(y, x\)
IsPartOf\(x, y\) $\rightarrow$ HasPart\(y, x\)
Real-world Coverage
Real-world Coverage

* We selected all rule sets from the MOWLCorp with less than 1000 rules of the form $A(x) \rightarrow \exists y. R(x, y) \land B(y)$
Real-world Coverage

* We selected all rule sets from the MOWLCorp with less than 1000 rules of the form $A(x) \rightarrow \exists y. R(x, y) \land B(y)$

* We also considered (all) the rule sets in the Oxford Ontology Library.
Real-world Coverage

* We selected all rule sets from the MOWLCorp with less than 1000 rules of the form \( A(x) \rightarrow \exists y \cdot R(x, y) \land B(y) \)

* We also considered (all) the rule sets in the Oxford Ontology Library.
Real-world Coverage

* We selected all rule sets from the MOWLCorp with less than 1000 rules of the form $A(x) \rightarrow \exists y. R(x, y) \land B(y)$

* We also considered (all) the rule sets in the Oxford Ontology Library. 

![Bar chart showing real-world coverage comparison between MFA (884) and RMFA (936) with +6% increase.](chart.png)
Real-world Coverage

* We selected all rule sets from the MOWLCorp with less than 1000 rules of the form $A(x) \rightarrow \exists y . R(x, y) \land B(y)$

* We also considered (all) the rule sets in the Oxford Ontology Library.

* We developed a cyclicity notion, i.e., sufficient condition for chase non-termination: Restricted Model-Faithful Cyclicity (RMFC)

![Bar graph showing MFA (884) and RMFA (936) with a +6% increase.]
We selected all rule sets from the MOWLCorpor with less than 1000 rules of the form $A(x) \rightarrow \exists y \cdot R(x, y) \land B(y)$.

We also considered (all) the rule sets in the Oxford Ontology Library.

We developed a cyclicity notion, i.e., sufficient condition for chase non-termination: Restricted Model-Faithful Cyclicity (RMFC).
Real-world Coverage

* We selected all rule sets from the MOWLCorp with less than 1000 rules of the form \( A(x) \rightarrow \exists y. R(x, y) \land B(y) \)

* We also considered (all) the rule sets in the Oxford Ontology Library.

* We developed a cyclicity notion, i.e., sufficient condition for chase non-termination: Restricted Model-Faithful Cyclicity (RMFC)
## Membership Checks

<table>
<thead>
<tr>
<th></th>
<th>RJA</th>
<th>RMSA</th>
<th>RMFA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No restrictions</td>
<td>ExpTime</td>
<td>ExpTime</td>
<td>2-ExpTime</td>
</tr>
<tr>
<td>Variables per rule bounded</td>
<td>P</td>
<td>P</td>
<td>2-ExpTime</td>
</tr>
</tbody>
</table>
## Membership Checks

<table>
<thead>
<tr>
<th></th>
<th>RJA</th>
<th>RMSA</th>
<th>RMFA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No restrictions</td>
<td>ExpTime</td>
<td>ExpTime</td>
<td>2-ExpTime</td>
</tr>
<tr>
<td>Variables per rule bounded</td>
<td>P</td>
<td>P</td>
<td>2-ExpTime</td>
</tr>
</tbody>
</table>

**BCQ entailment:** 2-ExpTime
Ensuring Tractability of the Chase
Ensuring Tractability of the Chase

Tractable Query Answering for Expressive Ontologies and Existential Rules

David Carral, Irina Dragoste, and Markus Krötzsch

[ISWC 2017]
Existential Dependency Graph

\[
\begin{align*}
A(x) & \rightarrow \exists y . S(x, y) \land B(y) \\
B(x) & \rightarrow \exists z . R(x, z) \land D(z) \\
D(x) & \rightarrow E(x) \\
E(x) & \rightarrow \exists w . R(x, w) \\
B(x) \land C(x) & \rightarrow E(x) \\
S(x, y) & \rightarrow C(x)
\end{align*}
\]
Existential Dependency Graph

\[
\begin{align*}
  A(x) &\rightarrow S(x, y(x)) \land B(y(x)) \\
  B(x) &\rightarrow R(x, z(x)) \land D(z(x)) \\
  D(x) &\rightarrow E(x) \\
  E(x) &\rightarrow R(x, w(x)) \\
  B(x) \land C(x) &\rightarrow E(x) \\
  S(x, y) &\rightarrow C(x)
\end{align*}
\]
Existential Dependency Graph

\[ A(x) \rightarrow S(x, y(x)) \land B(y(x)) \]
\[ B(x) \rightarrow R(x, z(x)) \land D(z(x)) \]
\[ D(x) \rightarrow E(x) \]
\[ E(x) \rightarrow R(x, w(x)) \]
\[ B(x) \land C(x) \rightarrow E(x) \]
\[ S(x, y) \rightarrow C(x) \]
Existential Dependency Graph

\[ A(x) \rightarrow S(x, y(x)) \land B(y(x)) \]
\[ B(x) \rightarrow R(x, z(x)) \land D(z(x)) \]
\[ D(x) \rightarrow E(x) \]
\[ E(x) \rightarrow R(x, w(x)) \]
\[ B(x) \land C(x) \rightarrow E(x) \]
\[ S(x, y) \rightarrow C(x) \]
Existential Dependency Graph

\[
\begin{align*}
A(x) & \rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) & \rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) & \rightarrow E(x) \\
E(x) & \rightarrow R(x, w(x)) \\
B(x) \land C(x) & \rightarrow E(x) \\
S(x, y) & \rightarrow C(x)
\end{align*}
\]

Reasoning over Existential Rules with Acyclicity Notions
David Carral
17/33
Existential Dependency Graph

\[
\begin{align*}
A(x) & \rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) & \rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) & \rightarrow E(x) \\
E(x) & \rightarrow R(x, w(x)) \\
B(x) \land C(x) & \rightarrow E(x) \\
S(x, y) & \rightarrow C(x) \\
A(c) & \\
S(c, y(c)), B(y(c)) & \\
R(y(c), z(y(c))), D(z(y(c))) & 
\end{align*}
\]
Existential Dependency Graph

\[
A(x) \rightarrow S(x, y(x)) \land B(y(x))
\]

\[
B(x) \rightarrow R(x, z(x)) \land D(z(x))
\]

\[
D(x) \rightarrow E(x)
\]

\[
E(x) \rightarrow R(x, w(x))
\]

\[
B(x) \land C(x) \rightarrow E(x)
\]

\[
S(x, y) \rightarrow C(x)
\]
Existential Dependency Graph

\[
\begin{align*}
A(x) & \rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) & \rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) & \rightarrow E(x) \\
E(x) & \rightarrow R(x, w(x)) \\
B(x) \land C(x) & \rightarrow E(x) \\
S(x, y(x)) & \rightarrow C(x)
\end{align*}
\]
A(x) → S(x, y(x)) ∧ B(y(x))
B(x) → R(x, z(x)) ∧ D(z(x))
D(x) → E(x)
E(x) → R(x, w(x))
B(x) ∧ C(x) → E(x)
S(x, y) → C(x)

B(c)
R(c, z(c)), D(z(c)),
E(z(c)),
R(z(c), w(z(c)))
Existential Dependency Graph

\[\begin{align*}
A(x) &\rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) &\rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) &\rightarrow E(x) \\
E(x) &\rightarrow R(x, w(x)) \\
B(x) \land C(x) &\rightarrow E(x) \\
S(x, y) &\rightarrow C(x)
\end{align*}\]
Existential Dependency Graph

\[ A(x) \rightarrow S(x, y(x)) \land B(y(x)) \]
\[ B(x) \rightarrow R(x, z(x)) \land D(z(x)) \]
\[ D(x) \rightarrow E(x) \]
\[ E(x) \rightarrow R(x, w(x)) \]
\[ B(x) \land C(x) \rightarrow E(x) \]
\[ S(x, y) \rightarrow C(x) \]
Existential Dependency Graph

\[ A(x) \rightarrow S(x, y(x)) \land B(y(x)) \]
\[ B(x) \rightarrow R(x, z(x)) \land D(z(x)) \]
\[ D(x) \rightarrow E(x) \]
\[ E(x) \rightarrow R(x, w(x)) \]
\[ B(x) \land C(x) \rightarrow E(x) \]
\[ S(x, y) \rightarrow C(x) \]
Existential Dependency Graph

\[
\begin{align*}
A(x) &\rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) &\rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) &\rightarrow E(x) \\
E(x) &\rightarrow R(x, w(x)) \\
B(x) \land C(x) &\rightarrow E(x) \\
S(x, y) &\rightarrow C(x)
\end{align*}
\]

\[
\begin{align*}
A(c) \\
S(c, y(c)), B(y(c)), \\
C(y(c)), \\
E(y(c)), \\
R(y(c), w(y(c)))
\end{align*}
\]
Existential Dependency Graph

\[
\begin{align*}
A(x) & \rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) & \rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) & \rightarrow E(x) \\
E(x) & \rightarrow R(x, w(x)) \\
B(x) \land C(x) & \rightarrow E(x) \\
S(x, y) & \rightarrow C(x)
\end{align*}
\]
Existential Dependency Graph

\[ A(x) \rightarrow S(x, y(x)) \land B(y(x)) \]
\[ B(x) \rightarrow R(x, z(x)) \land D(z(x)) \]
\[ D(x) \rightarrow E(x) \]
\[ E(x) \rightarrow R(x, w(x)) \]
\[ B(x) \land C(x) \rightarrow E(x) \]
\[ S(x, y) \rightarrow C(x) \]
Existential Dependency Graph

\[
\begin{align*}
A(x) & \rightarrow S(x, y(x)) \land B(y(x)) \\
B(x) & \rightarrow R(x, z(x)) \land D(z(x)) \\
D(x) & \rightarrow E(x) \\
E(x) & \rightarrow R(x, w(x)) \\
B(x) \land C(x) & \rightarrow E(x) \\
S(x, y) & \rightarrow C(x)
\end{align*}
\]
(a) Acyclicity
(a) Acyclicity
(a) Acyclicity

\[ w(z(c)) \]
(a) Acyclicity

\[ w \rightarrow z \rightarrow c \rightarrow y(w(z(c))) \]

\[ z(c) \rightarrow w(z(c)) \rightarrow y(w(z(c))) \]
(a) Acyclicity

\[
\begin{align*}
&y(w(z(c))) \\
&w(z(c)) \\
&z(c) \\
&z(y(w(z(c))))
\end{align*}
\]
(a) Acyclicity

\[ w(z(c)) \]

\[ y(w(z(c))) \]

\[ w(z(y(w(z(c)))))) \]

\[ z(y(w(z(c)))) \]
(a) Acyclicity

\[ z(c) \]
\[ y(w(z(c))) \]
\[ w(z(c)) \]
\[ w(z(y(w(z(c))))) \]
\[ z(y(w(z(c))))) \]

\[ w(z(y(w(z(c))))) \]

\[ z(y(w(z(c))))) \]

\[ w(z(y(w(z(c))))) \]

\[ ........ \]
(a) Acyclicity

Remark. If the existential dependency graph of a given set of rules is acyclic, then the set of terms introduced during the computation of the chase is finite.
(f) Arity at Most 1

\[
\text{Film}(x) \rightarrow \exists y . \text{IsFilmDirectedBy}(x, y) \land \text{Director}(y) \\
A(x) \land B(x, w) \land C(x, z) \rightarrow \exists z . R(x, w, z)
\]
(f) Arity at Most 1

\[
\begin{align*}
\text{Film}(x) & \rightarrow \exists y . \text{IsFilmDirectedBy}(x, y) \land \text{Director}(y) \\
\text{Film}(x) & \rightarrow \text{IsFilmDirectedBy}(x, y(x)) \land \text{Director}(y(x)) \\
A(x) \land B(x, y) \land C(x, z) & \rightarrow \exists z . \text{R}(x, y(z)) \\
A(x) \land B(x, w) \land C(x, z) & \rightarrow \text{R}(x, w, z(x, w))
\end{align*}
\]
(f) Arity at Most 1

\[ \text{Film}(x) \rightarrow \exists y \cdot \text{IsFilmDirectedBy}(x, y) \land \text{Director}(y) \]
\[ \text{Film}(x) \rightarrow \text{IsFilmDirectedBy}(x, y(x)) \land \text{Director}(y(x)) \]
\[ A(x) \land B(x, y) \land C(x, z) \rightarrow \exists z \cdot R(x, y, z) \]
\[ A(x) \land B(x, w) \land C(x, z) \rightarrow R(x, w, z(x, w)) \]

Remark. If the arity of every function symbol in the skolemisation of a program is at most 1, then every term in the chase is of the form \( x_1(\ldots x_n(c)\ldots) \) with \( c \) constant.
(f) Arity at Most 1

Remark. If the arity of every function symbol in the skolemisation of a program is at most 1, then every term in the chase is of the form $x_1(...x_n(c)...) \text{ with } c \text{ constant.}$
(f) Arity at Most 1

**Remark.** If the arity of every function symbol in the skolemisation of a program is at most 1, then every term in the chase is of the form $x_1(...x_n(c)...) \text{ with } c \text{ constant.}$

**Corollary.** Every term occurring in the chase corresponds to a path in the dependency graph and a constant.
(f) Arity at Most 1

**Remark.** If the arity of every function symbol in the skolemisation of a program is at most 1, then every term in the chase is of the form $x_1(\ldots x_n(c)\ldots)$ with $c$ constant.

**Corollary.** Every term occurring in the chase corresponds to a path in the dependency graph and a constant.
(f) Arity at Most 1

**Remark.** If the arity of every function symbol in the skolemisation of a program is at most 1, then every term in the chase is of the form $x_1(...x_n(c)...) \text{ with } c \text{ constant.}$

**Corollary.** Every term occurring in the chase corresponds to a path in the dependency graph and a constant.
Ensuring Tractability
Ensuring Tractability

(a) The dependency graph is acyclic.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

All skolem terms correspond to some path in the dependency graph and some constant.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

(w) The number of variables per rule is bounded.

All skolem terms correspond to some path in the dependency graph and some constant.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

All skolem terms correspond to some path in the dependency graph and some constant.

(w) The number of variables per rule is bounded.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

All skolem terms correspond to some path in the dependency graph and some constant.

(w) The number of variables per rule is bounded.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.

The number of paths in the dependency graph is polynomial.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

(w) The number of variables per rule is bounded.

All skolem terms correspond to some path in the dependency graph and some constant.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.

The number of paths in the dependency graph is polynomial.

Polynomiality
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

(w) The number of variables per rule is bounded.

Polynomiality

All skolem terms correspond to some path in the dependency graph and some constant.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.

The number of paths in the dependency graph is polynomial.

(?)
Braids
Braids
Braids
Braids
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

(w) The number of variables per rule is bounded.

Polynomiality

All skolem terms correspond to some path in the dependency graph and some constant.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.

The number of paths in the dependency graph is polynomial.

Polynomiality

(?)
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

(w) The number of variables per rule is bounded.

All skolem terms correspond to some path in the dependency graph and some constant.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.

The number of paths in the dependency graph is polynomial.

(b) The length of the braids in the dependency graph is bounded.
Ensuring Tractability

(a) The dependency graph is acyclic.

(f) The arity of all function symbols in the skolemisation of the program is at most 1.

(w) The number of variables per rule is bounded.

Polynomiality

All skolem terms correspond to some path in the dependency graph and some constant.

Rules can be applied in polynomial time.

The number of facts is polynomial in the number of terms.

The number of paths in the dependency graph is polynomial.

(b) The length of the braids in the dependency graph is bounded.

Caveats.
1. Fixed query size.
2. Horn rule set.
Real-world coverage: SRI Ontologies

\[ A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \quad \Leftrightarrow \quad \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x) \]

\[ A \sqsubseteq B_1 \sqcup \ldots \sqcup B_n \quad \Rightarrow \quad A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x) \]

\[ A \sqsubseteq \forall R \cdot B \quad \Rightarrow \quad A(y) \land R(x, y) \rightarrow B(x) \]

\[ A \sqsubseteq \exists R \cdot B \quad \Rightarrow \quad A(x) \rightarrow \exists y \cdot R(x, y) \land B(y) \]

\[ R \sqsubseteq S \quad \Rightarrow \quad R(x, y) \rightarrow S(x, y) \]

\[ R \circ S \sqsubseteq V \quad \Rightarrow \quad R(x, y) \land S(y, z) \rightarrow S(x, z) \]

\[ R_1 \sqcap \ldots \sqcap R_n \sqsubseteq S \quad \Rightarrow \quad \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y) \]

\[ A(a) \quad \Rightarrow \quad A(a) \]

\[ R(a, b) \quad \Rightarrow \quad R(a, b) \]
Real-world coverage: SRI Ontologies

\[
A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \iff \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x)
\]

\[
A \sqsubseteq B_1 \sqcup \ldots \sqcup B_n \iff A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x)
\]

\[
A \sqsubseteq \forall R . B \iff A(y) \land R(x, y) \rightarrow B(x)
\]

\[
A \sqsubseteq \exists R . B \iff A(x) \rightarrow \exists y . R(x, y) \land B(y)
\]

\[
R \sqsubseteq S \iff R(x, y) \rightarrow S(x, y)
\]

\[
R \circ S \sqsubseteq V \iff R(x, y) \land S(y, z) \rightarrow S(x, z)
\]

\[
R_1 \sqcap \ldots \sqcap R_n \sqsubseteq S \iff \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y)
\]

\[
A(a) \iff A(a)
\]

\[
R(a, b) \iff R(a, b)
\]

**Remark 1.** Deciding CQ entailment for SRI ontologies is 2ExpTime-Hard and in 3ExpTime.
Real-world coverage: SRI Ontologies

\[ A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \implies \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x) \]
\[ A \sqsubseteq B_1 \sqcup \ldots \sqcup B_n \implies A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x) \]
\[ A \sqsubseteq \forall R \cdot B \implies A(y) \land R(x, y) \rightarrow B(x) \]
\[ A \sqsubseteq \exists R \cdot B \implies A(x) \rightarrow \exists y \cdot R(x, y) \land B(y) \]
\[ R \sqsubseteq S \implies R(x, y) \rightarrow S(x, y) \]
\[ R \circ S \sqsubseteq V \implies R(x, y) \land S(y, z) \rightarrow S(x, z) \]
\[ R_1 \sqcap \ldots \sqcap R_n \sqsubseteq S \implies \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y) \]

**Remark 1.** Deciding CQ entailment for SRI ontologies is 2ExpTime-Hard and in 3ExpTime.

**Remark 2.**
1. SRI rules feature at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology is of arity one.
Real-world coverage: SRI Ontologies

\[ A_1 \sqcap \ldots \sqcap A_n \subseteq B \quad \Rightarrow \quad \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x) \]

\[ A \subseteq B_1 \sqcup \ldots \sqcup B_n \quad \Rightarrow \quad A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x) \]

\[ A \subseteq \forall R \cdot B \quad \Rightarrow \quad A(y) \land R(x, y) \rightarrow B(x) \]

\[ A \subseteq \exists R \cdot B \quad \Rightarrow \quad A(x) \rightarrow \exists y \cdot R(x, y) \land B(y) \]

\[ R \subseteq S \quad \Rightarrow \quad R(x, y) \rightarrow S(x, y) \]

\[ R \circ S \subseteq V \quad \Rightarrow \quad R(x, y) \land S(y, z) \rightarrow S(x, z) \]

\[ R_1 \sqcap \ldots \sqcap R_n \subseteq S \quad \Rightarrow \quad \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y) \]

\[ A(a) \quad \Rightarrow \quad A(a) \]

\[ R(a, b) \quad \Rightarrow \quad R(a, b) \]

**Remark 1.** Deciding CQ entailment for SRI ontologies is 2ExpTime-Hard and in 3ExpTime.

**Remark 2.**
1. SRI rules feature at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology is of arity one.
Real-world coverage: SRI Ontologies

\[ A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \leftrightarrow \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x) \]

\[ A \sqsubseteq B_1 \sqcup \ldots \sqcup B_n \leftrightarrow A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x) \]

\[ A \sqsubseteq \forall R . B \leftrightarrow A(y) \land R(x, y) \rightarrow B(x) \]

\[ A \sqsubseteq \exists R . B \leftrightarrow A(x) \rightarrow \exists y . R(x, y) \land B(y) \]

\[ R \sqsubseteq S \leftrightarrow R(x, y) \rightarrow S(x, y) \]

\[ R \circ S \sqsubseteq V \leftrightarrow R(x, y) \land S(y, z) \rightarrow S(x, z) \]

\[ R_1 \sqcap \ldots \sqcap R_n \sqsubseteq S \leftrightarrow \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y) \]

\[ A(a) \leftrightarrow A(a) \]

\[ R(a, b) \leftrightarrow R(a, b) \]

Remark 1. Deciding CQ entailment for SRI ontologies is 2ExpTime-Hard and in 3ExpTime.

Remark 2.
1. SRI rules feature at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology is of arity one.
Real-world coverage: SRI Ontologies

\[ A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \iff \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x) \]

\[ A \sqsubseteq B_1 \sqcup \ldots \sqcup B_n \iff A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x) \]

\[ A \sqsubseteq \forall R . B \iff A(y) \land R(x, y) \rightarrow B(x) \]

\[ A \sqsubseteq \exists R . B \iff A(x) \rightarrow \exists y . R(x, y) \land B(y) \]

\[ R \sqsubseteq S \iff R(x, y) \rightarrow S(x, y) \]

\[ R \circ S \sqsubseteq V \iff R(x, y) \land S(y, z) \rightarrow S(x, z) \]

\[ R_1 \sqcap \ldots \sqcap R_n \sqsubseteq S \iff \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y) \]

\[ A(a) \iff A(a) \]

Remark 1. Deciding CQ entailment for SRI ontologies is 2ExpTime-Hard and in 3ExpTime.

Remark 2.
1. SRI rules feature at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology is of arity one.
Real-world coverage: SRI Ontologies

\[
\begin{align*}
A_1 \cap \ldots \cap A_n \subseteq B & \implies \bigwedge_{i=1}^{n} A_i(x) \rightarrow B(x) \\
A \subseteq B_1 \cup \ldots \cup B_n & \implies A(x) \rightarrow \bigvee_{i=1}^{n} B_i(x) \\
A \subseteq \forall R . B & \implies A(y) \land R(x, y) \rightarrow B(x) \\
A \subseteq \exists R . B & \implies A(x) \rightarrow \exists y . R(x, y) \land B(y) \\
R \subseteq S & \implies R(x, y) \rightarrow S(x, y) \\
R \circ S \subseteq V & \implies R(x, y) \land S(y, z) \rightarrow S(x, z) \\
R_1 \cap \ldots \cap R_n \subseteq S & \implies \bigwedge_{i=1}^{n} R_i(x, y) \rightarrow S(x, y) \\
A(a) & \implies A(a) \\
R(a, b) & \implies R(a, b)
\end{align*}
\]

Remark 1. Deciding CQ entailment for SRI ontologies is 2ExpTime-Hard and in 3ExpTime.

Remark 2.
1. SRI rules feature at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology is of arity one.
SRI Axioms

Remark 2.
1. Every rule in an SRI ontology has at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology has arity one
SRI Axioms

**Remark 2.**
1. Every rule in an SRI ontology has at most 3 variables.
2. Every function symbol in the skolemisation of a SRI ontology has arity one.

**Corollary.** To guarantee that tractable CQ entailment over a SRI ontology is possible we only need to verify the following:
1. Acyclicity.
2. Braid length in the dependency graph is bounded.
Evaluation Results
## Evaluation Results

### Acyclicity

<table>
<thead>
<tr>
<th></th>
<th>MOWL Corpus</th>
<th>Oxford Ontology Repo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ontologies</td>
<td>1576</td>
<td>225</td>
</tr>
<tr>
<td>Acyclic</td>
<td>974 (61.8%)</td>
<td>170 (75.6%)</td>
</tr>
</tbody>
</table>
Evaluation Results

Acyclicity

<table>
<thead>
<tr>
<th></th>
<th>MOWL Corpus</th>
<th>Oxford Ontology Repo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ontologies</td>
<td>1576</td>
<td>225</td>
</tr>
<tr>
<td>Acyclic</td>
<td>974 (61.8%)</td>
<td>170 (75.6%)</td>
</tr>
</tbody>
</table>

Braid Length

<table>
<thead>
<tr>
<th>(max. length of a braid)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>11</th>
<th>22</th>
<th>23</th>
<th>25</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>(count)</td>
<td>851</td>
<td>153</td>
<td>56</td>
<td>61</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>1144</td>
</tr>
<tr>
<td></td>
<td>74</td>
<td>88</td>
<td>93</td>
<td>98</td>
<td>99</td>
<td>99</td>
<td>99.1</td>
<td>99.3</td>
<td>99.9</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
More Results!
VLog
VLog

Efficient Model Construction for Horn Logic with VLog — System Description
Jacopo Urbani, Markus Krötzsch, Ceriel J. H. Jacobs, Irina Dragoste, and David Carral
[IJCAR 2018]
An Implementation for the DF Restricted Chase

Consider a rule set $R$ and an instance $I$.
Let $R_{\forall}$ and $R_{\exists} = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
An Implementation for the DF Restricted Chase

Consider a rule set $R$ and an instance $I$.

Let $R_{\forall}$ and $R_{\exists} = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.

$I = F_1$
Consider a rule set $R$ and an instance $I$.

Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
An Implementation for the DF Restricted Chase

Consider a rule set $R$ and an instance $I$. Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
An Implementation for the DF Restricted Chase

Consider a rule set $R$ and an instance $I$.
Let $R_{\forall}$ and $R_{\exists} = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively.
The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
An Implementation for the DF Restricted Chase

Consider a rule set $R$ and an instance $I$.
Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
Consider a rule set $R$ and an instance $I$.

Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively.

The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.

$$I = F_1 \xrightarrow{R^*_\forall} G_1 \xrightarrow{r_1} G_{11} \xrightarrow{r_2} G_{12} \xrightarrow{r_n} G_{1n} \xrightarrow{R^*_\forall} F_2 \xrightarrow{r_1} G_2 \xrightarrow{r_2} G_{21} \xrightarrow{r_n} G_{22} \xrightarrow{\ldots} G_{2n}$$
Consider a rule set $R$ and an instance $I$.

Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively.

The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
Consider a rule set $R$ and an instance $I$. Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
Consider a rule set $R$ and an instance $I$. Let $R_\forall$ and $R_\exists = \{r_1, \ldots, r_n\}$ be the sets of all Datalog and non-Datalog rules in $R$, respectively. The Datalog-first restricted chase of $R$ and $I$, denoted with $\text{Ch}(R, I)$, is computed as follows.
Performance: VLog vs RDFox

**Fig. 1.** Memory usage (left) and materialisation time (right) for VLog and RDFox
Conclusions
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:
  * Experiments with non-DL existential rule sets
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:
* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
* Efficient implementation for acyclicity checks
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
* Efficient implementation for acyclicity checks
* Optimise chase reasoners
Problem Solved?

This is it, everybody should use existential rules + acyclicity notions!
Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge engineers in “repairing” non-acyclic rule sets
* Efficient implementation for acyclicity checks
* Optimise chase reasoners
* The disjunctive case
Problem Solved?

This is it, everybody should use existential rules +
acyclicity notions!

Surely not, a lot of work still needs to be done:

* Experiments with non-DL existential rule sets
* Develop tools that can assist knowledge
  engineers in “repairing” non-acyclic rule sets
* Efficient implementation for acyclicity checks
* Optimise chase reasoners
* The disjunctive case
Bibliography
Reasoning over Existential Rules with Acyclicity Notions and the Datalog-first Restricted Chase

David Carral

Slides available at https://iccl.inf.tu-dresden.de/web/Existential-rules-acyclicity
Bibliography: Sections

* **First section**
  Restricted Chase (Non)Termination for Existential Rules with Disjunctions [IJCAI 2017]
  https://iccl.inf.tu-dresden.de/web/Inproceedings3140/en

* **Second section**
  Tractable Query Answering for Expressive Ontologies and Rules [ISWC 2017]
  https://iccl.inf.tu-dresden.de/web/Inproceedings3163/en

* **Third section**
  Efficient Model Construction for Horn Logic with VLog [IJCAR 2018]
  https://iccl.inf.tu-dresden.de/web/Article3046/en
Bibliography: Rule Engines

* **VLog**
  Efficient Model Construction for Horn Logic with VLog [IJCAR 2018]
  [https://iccl.inf.tu-dresden.de/web/Article3046/en](https://iccl.inf.tu-dresden.de/web/Article3046/en)
  Column-Oriented Datalog Materialization for Large Knowledge Graphs [AAAI 2016]

* **RDFox**
  Parallel Materialisation of Datalog Programs in Centralised, Main-Memory RDF Systems [AAAI 2014]
Bibliography: Acyclicity Notions

* **Restricted Model-Faithful Acyclicity** (RMFA)
  Restricted Chase (Non)Termination for Existential Rules with Disjunctions. [IJCAI 2017]
  https://iccl.inf.tu-dresden.de/web/Inproceedings3140/en

* **Model-Faithful Acyclicity** (MFA)
  https://iccl.inf.tu-dresden.de/web/Article4005/en

* **Joint Acyclicity** (JA)
  Extending Decidable Existential Rules by Joining Acyclicity and Guardedness [IJACI 2011]
  https://iccl.inf.tu-dresden.de/web/Inproceedings3149/en

* **Weak Acyclicity** (WA)