Query Rewriting Under Existential Rules

Andreas Pieris

School of Informatics, University of Edinburgh, UK

based on joint work with Pablo Barceló, Gerald Berger, Andrea Calì, Georg Gottlob, Marco Manna, Giorgio Orsi and Pierfrancesco Veltri

DL Workshop, Montpellier, France, July 18 - 21, 2017
this talk is about **first-order rewritability** under the **basic decidable classes of existential rules**
Ontology-Based Query Answering

Certain-Answers\( (q, D, O) \) = \{ (c_1, \ldots, c_n) \in \text{dom}(D)^n \mid D \land O \models q(c_1, \ldots, c_n) \}
Ontology-Mediated Queries

$S$-database (ABox)

$Q = (S, O, q(x_1, \ldots, x_n))$

$Q(D) = \text{Certain-Answers}(q, D, O)$
Scalability in OMQ Evaluation

Exploit standard RDBMSs - efficient technology for answering queries
Query Rewriting

\[ Q = (S, O, q(x_1, \ldots, x_n)) \]

\[ \text{rewrite} \]

\[ Q_{\text{rew}}(x_1, \ldots, x_n) \]

a query that can be executed by a standard DBMS - first-order query

for every \textbf{S}-database \( D \) : \( Q(D) = Q_{\text{rew}}(D) \)

Query Rewriting: An Example

\[
\{ \forall x \, (\text{Person}(x) \rightarrow \exists y \, \text{HasFather}(x,y) \land \text{Person}(y)) \} \equiv \text{Person} \subseteq \exists \text{HasFather}.\text{Person}
\]

\[
\{ \text{Person}(\cdot), \text{HasFather}(\cdot,\cdot) \}
\]

\[
Q = (S, O, q())
\]

\[
Q_{\text{rew}} = \exists x \, \text{Person}(x) \land \text{HasFather}(\text{John},x) \lor \text{Person}(\text{John})
\]
First-Order Rewritability (FO-Rewritability)

\( \text{(OL,QL)} \)

- an ontology language (fragment of first-order logic)
- a database query language (sublanguage of first-order queries)

**Definition:** An OMQ language \( O \) is **FO-Rewritable** if every \( Q \in O \) is FO-Rewritable.
FO-Rewritability: The Main Questions

1. Can we isolate meaningful OMQ languages that are FO-Rewritable?

2. For non-FO-Rewritable languages, can we decide FO-Rewritability?

3. What is the size of the FO rewritings? Can we do better?

...have been extensively studied for DL- and rule-based OMQ languages
Existential Rules

(a.k.a. tuple-generating dependencies)

\[
\forall x \forall y (\varphi(x,y) \rightarrow \exists z \psi(x,z))
\]

\[
\forall x (\text{Person}(x) \rightarrow \exists y \text{HasFather}(x,y) \land \text{Person}(y)) \equiv \text{Person} \subseteq \exists \text{HasFather}.\text{Person}
\]

\[
\forall x \forall y (\text{HasChild}(x,y) \land \text{Human}(y) \rightarrow \text{Human}(x)) \equiv \exists \text{HasChild}.\text{Human} \subseteq \text{Human}
\]
Existential Rules

(a.k.a. tuple-generating dependencies)

\[ \varphi(x, y) \rightarrow \exists z \, \psi(x, z) \]

\[
\text{Person}(x) \rightarrow \exists y \, \text{HasFather}(x, y), \text{Person}(y) \equiv \text{Person} \sqsubseteq \exists \text{HasFather}.\text{Person}
\]

\[
\text{HasChild}(x, y), \text{Human}(y) \rightarrow \text{Human}(x) \equiv \exists \text{HasChild}.\text{Human} \sqsubseteq \text{Human}
\]
Existential Rules

(a.k.a. tuple-generating dependencies)

\[ \varphi(x,y) \rightarrow \exists z \psi(x,z) \]

(∃Rules,CQ)
Guardedness

**Frontier-Guarded**
one body-atom contains all
the $\forall$-variables in the head

**Guarded**
one body-atom contains
all the $\forall$-variables

**Linear**
one body-atom

$R(x), \varphi(x,y) \rightarrow \exists z \psi(x,z)$


$R(x,y), \varphi(x,y) \rightarrow \exists z \psi(x,z)$


$R(x,y) \rightarrow \exists z \psi(x,z)$

Acyclicity

(...or, non-recursive - the predicate graph is acyclic)

\[ R(x, y), R(y, z) \rightarrow \exists w \ P(x), S(x, w) \]

\[ T(x) \rightarrow P(x) \]
Stickiness

(...or, do not forget the joins)

\[ R(x, y), P(y, z) \rightarrow \exists w \ T(x, y, w) \]

\[ T(x, y, z) \rightarrow \exists w \ S(y, w) \]

\[ \]

\[ R(x_1, \ldots, x_n), P(y_1, \ldots, y_m) \rightarrow T(x_1, \ldots, x_n, y_1, \ldots, y_m) \]

Classes of Existential Rules

(a.k.a. Datalog\(\pm\) languages)
Classes of Existential Rules

(a.k.a. Datalog± languages)

What about FO-Rewritability?
Classes of Existential Rules

(a.k.a. Datalog± languages)

Dangerous zone!
Guardedness and FO-Rewritability

Theorem: (Guarded, CQ) is not FO-Rewritable

\[ Q = (\{P, R\}, \{R(x,y), P(y) \rightarrow P(x)\}, P(c_n)) \]

\[ D \supseteq \{P(c_1)\}, \text{ and contains no other P-atom} \]

\( Q_{\text{rew}} \) has to check for the existence of an \( R \)-path in \( D \) of unbounded length

\[
\begin{align*}
c_n & \xrightarrow{R} #_{n-1} & \xrightarrow{R} #_{n-2} & \cdots & \xrightarrow{R} #_2 & \xrightarrow{R} c_1
\end{align*}
\]

compute the transitive closure of \( R \) - not possible via a first-order query
Theorem: \((L, CQ)\), where \(L \in \{ \text{Linear, Acyclic, Sticky} \}\), is FO-Rewritable via the Bounded Derivation Depth Property (BDDP)
**Definition:** \((\mathbf{L}, \mathbf{CQ})\) enjoys the BDDP if:

for every \(Q = (S, O, q) \in (\mathbf{L}, \mathbf{CQ})\), there exists \(\delta \geq 0\) such that,

for every \(S\)-database \(D\), \(Q(D) = q(\text{chase}^\delta(D, O))\)

Bounded Derivation Depth Property (BDDP)

**Proposition:** BDDP $\Rightarrow$ FO-Rewritability

Each atom is obtained by at most $\beta$ atoms

$\Rightarrow$ to entail a CQ $q$ we need at most $|q| \cdot \beta^\delta$ database atoms
Bounded Derivation Depth Property (BDDP)

**Proposition:** BDDP $\Rightarrow$ FO-Rewritability

Given an OMQ $(S, O, q)$:

- $D_{\beta,\delta,q}$ be the set of all possible $S$-databases of size at most $|q| \cdot \beta^\delta$

- $C = \{ D \in D_{\beta,\delta,q} \mid q(\text{chase}(D,O)) \text{ is non-empty} \}$

- Convert $C$ into a UCQ

...in fact, the other direction also holds - FO-Rewritability $\Leftrightarrow$ BDDP
Theorem: \((L, CQ)\), where \(L \in \{\text{Linear, Acyclic, Sticky}\}\), is FO-Rewritable

Via the Bounded Derivation Depth Property (BDDP)

but, the BDDP-based algorithm is very expensive

can we do better?

Perfect Reformulation

Algorithm PerfectRef \((q, T)\)

**Input:** conjunctive query \(q\), TBox \(T\)

**Output:** union of conjunctive queries \(PR\)

\(PR := \{q\}\);

repeat

\(PR' := PR;\)

for each \(q \in PR'\) do

(a) for each \(g\) in \(q\) do

for each PI \(I\) in \(T\) do

if \(I\) is applicable to \(g\) then \(PR := PR \cup \{q[g/gr(g, I)]\}\)

(b) for each \(g_1, g_2\) in \(q\) do

if \(g_1\) and \(g_2\) unify then \(PR := PR \cup \{\tau(\text{reduce}(q, g_1, g_2))\}\)

until \(PR' = PR\);

return \(PR\)

Fig. 2 The algorithm PerfectRef

Applicability → Soundness

Reduction → Completeness


rewriting step

reduction step
Perfect Reformulation for Existential Rules

\[ R(y,x), P(y) \rightarrow \exists z \ T(z,x,x) \]
\[ \exists u \exists v \exists w \ T(u,v,w), P(w) \]
\[ g = \{ u \rightarrow z, v \rightarrow x, w \rightarrow x \} \]
\[ T(z,x,x) \]

thus, we can simulate a chase step by applying a backward resolution step

\[ \exists u \exists v \exists w \ T(u,v,w), P(w) \lor \exists x \exists y \ R(y,x), P(y), P(x) \]
Perfect Reformulation for Existential Rules

\[ R(y,x), P(y) \rightarrow \exists z \ T(z,x,x) \]

\[ \exists u \exists v \exists w \ T(u,v,w), P(u) \]

\[ \exists x \exists y \exists u \ R(x,y), P(x), P(u) \]

\[ g = \{ u \rightarrow z, v \rightarrow x, w \rightarrow x \} \]

thus, we can simulate a chase step by applying a backward resolution step.

unsound rewriting
Perfect Reformulation for Existential Rules

\[ R(y, x), P(y) \rightarrow \exists z \ T(z, x, x) \]

\[ \exists u \exists v \exists w \ T(u, v, w), P(u) \]

\[ g = \{ u \rightarrow z, v \rightarrow x, w \rightarrow x \} \]

**Applicability condition:** constants, join variables and free variables in the query do **NOT** unify with \( \exists \)-variables

...but, it may destroy completeness
Perfect Reformulation for Existential Rules

\[ R(y,x), P(y) \rightarrow \exists z \ T(z,x,x) \quad \exists u \exists v \exists w \ T(u,v,w), P(u) \]

\[ T(x,y,z) \rightarrow P(x) \]

\[ \exists u \exists v \exists w \ T(u,v,w), P(u) \lor \]

\[ \exists u \exists v \exists w \exists y \exists z \ T(u,v,w), T(u,y,z) \lor \]

(by the reduction step) \[ \exists u \exists v \exists w \ T(u,v,w) \lor \]

(by the rewriting step) \[ \exists x \exists y \ R(x,y), P(x) \]
applicability condition for existential rules

apply only useful reduction steps

FO-Rewritable OMQ Languages

Theorem: \((L, CQ)\), where \(L \in \{\text{Linear, Acyclic, Sticky}\}\), is FO-Rewritable via the Bounded Derivation Depth Property (BDDP).

but, the BDDP-based algorithm is very expensive. Can we do better?

use the XRewrite algorithm

Piece-based rewriting - based on a refined notion of unification

[König, Leclère, Mugnier & Thomazo, RR 2012, Semantic Web 2015]
Recap

What about deciding FO-Rewritability?

- Frontier-Guarded
- Guarded
- Linear
- Acyclic
- Sticky

- FO-Rewritable
- non-FO-Rewritable
Deciding FO-Rewritability

\[ Q = (S, O, q(x,y)) \]

\[ \{ R(x,y), S(y) \rightarrow S(x), \ R(x,y), P(x) \rightarrow S(y) \} \]

\[ \{ P(\cdot), R(\cdot, \cdot), S(\cdot) \} \]

\[ P(x) \land R(x, y) \land S(y) \]

\[ P(x) \land R(x, y) \land S(y) \]

\[ \text{rewrite} \]

\[ P(x) \land R(x, y) \]
Deciding FO-Rewritability

\[
Q = (S, O, q(y))
\]

\[
\{ R(x,y), S(y) \rightarrow S(x), \quad R(x,y), P(x) \rightarrow S(y) \}\\
\{ P(\cdot), R(\cdot, \cdot), S(\cdot) \}
\]

rewrite

\[\times\]
Deciding FO-Rewritability

\[
\text{FORew(L,QL)}
\]

**Input:** an OMQ \( Q \in (L,QL) \)

**Question:** is \( Q \) FO-Rewritable?

What is the complexity of \( \text{FORew(Guarded,CQ)} \) and \( \text{FORew(Frontier-Guarded,CQ)} \)?
Deciding FO-Rewritability

\[ \text{FORew}(L, QL) \]

Input: an OMQ \( Q \in (L, QL) \)

Question: is \( Q \) FO-Rewritable?

Theorem: \( \text{FORew}(L, CQ) \), where \( L \in \{ \text{Guarded, Frontier-Guarded} \} \) is in 3EXPTIME, and 2EXPTIME-hard even for bounded arity

[Barceló, Berger & P., 2017]
Deciding FO-Rewritability

Theorem: $\text{FORew}(\text{Guarded}, \text{BCQ})$ is in 3EXPTIME and 2EXPTIME-hard even for bounded arity

Upper Bound:
- Characterize FO-Rewritability via the finiteness of a set of certain "tree-like" databases
- Construct an alternating tree automaton $A$, with double-exponentially many states, such that the OMQ is FO-Rewritable iff the language of $A$ is finite

Lower Bound:
- Inherited from $\text{FORew}(\text{ELI}, \text{CQ})$
  [Bienvenu, Hansen, Lutz & Wolter, IJCAI 2016]
Tree Decomposition

\[ D = \{ R(a,b,c), T(c,e), R(b,c,d), S(c,d,a), P(d,f), T(f,f) \} \]
Tree Decomposition

\[ D = \{ R(a,b,c), T(c,e), R(b,c,d), S(c,d,a), P(d,f), T(f,f) \} \]
Tree Decomposition

$$D = \{ R(a,b,c), T(c,e), R(b,c,d), S(c,d,a), P(d,f), T(f,f) \}$$
Tree Decomposition

\[ D = \{ R(a,b,c), T(c,e), R(b,c,d), S(c,d,a), P(d,f), T(f,f) \} \]
Tree Decomposition

\[ D = \{ \text{R}(a,b,c), \text{T}(c,e), \text{R}(b,c,d), \text{S}(c,d,a), \text{P}(d,f), \text{T}(f,f) \} \]
C-Tree Databases

(...or, almost “tree-like” databases)

**Definition:** An S-database $D$ is a C-tree, where $C \subseteq D$, if it has the form:

$$
T_0, C \\
T_1, A_1 \\
T_2, A_2 \\
T_3, A_3 \\
T_4, A_4 \\
T_5, A_5
$$

for each $i > 0$, $|T_i| \leq \text{arity}(S)$
Characterizing FO-Rewritability

**Proposition:** Let $Q = (S, O, q) \in (\text{Guarded}, \text{BCQ})$:

\[ Q \text{ is FO-Rewritable} \]

unravelling and compactness \[\iff\] Q is UCQ-Rewritable

there exists $k \geq 0$ such that, for every C-tree $D$ over $S$, with $|\text{dom}(C)| \leq (\text{arity}(S, O) \cdot |q|)$, it holds that:

$D \vdash Q \Rightarrow$ there exists $D' \subseteq D$ with $|D'| \leq k$ such that $D' \not\vdash Q$
Characterizing FO-Rewritability

**Proposition:** Let $Q = (S, O, q) \in (\text{Guarded}, \text{BCQ})$:

$Q$ is FO-Rewritable

\[\Downarrow\]

there exist finitely many (non-isomorphic) $C$-trees $D$ over $S$, with $|\text{dom}(C)| \leq (\text{arity}(S,O) \cdot |q|)$, such that:

(i) $D \models Q$

(ii) remove an atom from $D \Rightarrow Q$ is violated

(iii) $D$ is non-redundant
Well-Colored Tree Decomposition

\[ D = \{ R(a,b,c), T(c,e), R(b,c,d), S(c,d,a), P(d,f), T(f,f) \} \]

- node \( v \) is red \( \Rightarrow \) \( v \) is the least common ancestor of a non-empty set of blue nodes
Characterizing FO-Rewritability

**Proposition:** Let $Q = (S, O, q) \in \text{(Guarded, BCQ)}$:

$Q$ is FO-Rewritable

\[ \Downarrow \]

there exist finitely many (non-isomorphic) $C$-trees $D$ over $S$, with $|\text{dom}(C)| \leq (\text{arity}(S, O) \cdot |q|)$, such that:

(i) $D \models Q$

(ii) remove an atom from $D \Rightarrow Q$ is violated

(iii) $D$ is well-colored

the language of an alternating tree automaton $A$ with double-exponentially many states
Characterizing FO-Rewritability

**Proposition:** Let $Q = (S, O, q) \in (\text{Guarded}, \text{BCQ})$: 

$Q$ is FO-Rewritable

\[\iff\]

the language of $A$ is finite

(which is feasible in exponential time in the number of states)
Deciding FO-Rewritability

Theorem: \( \text{FORew(Frontier-Guarded,BCQ)} \) is in \( 3\text{EXPTIME} \)

\[ Q \in (\text{Frontier-Guarded,BCQ}) \]

\[ Q' \in (\text{Frontier-Guarded,BAQ}) \]

\[ Q'' \in (\text{Guarded,BAQ}) \]

- a BCQ is a frontier-guarded rule
- by treeifying the rule-bodies [Bárán, ten Cate & Segoufin, ICALP 2011, J. ACM 2015]

Q is FO-Rewritable \( \Leftrightarrow \) Q" is FO-Rewritable

[Barceló, Berger & P., 2017]
Deciding FO-Rewritability: Next Steps

• Practical rewriting algorithms for \((\text{Frontier-Guarded}, \text{CQ})\)

• Such a practical algorithm exists for \((\text{EL}, \text{AQ})\)
  [Hansen, Lutz, Seylan & Wolter, IJCAI 2015]

• …and it has been recently extended to \((\text{EL}, \text{CQ})\)
  [Hansen & Lutz, DL 2017]
Recap

What about the size of the FO rewritings?

can be checked in 3EXPTIME
Height/Size of XR rewrite (Q)

Given an OMQ \( Q = (S, O, q) \in (L, CQ) \)

<table>
<thead>
<tr>
<th>L</th>
<th>Height</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>(</td>
<td>q</td>
</tr>
<tr>
<td>Acyclic</td>
<td>(</td>
<td>q</td>
</tr>
<tr>
<td>Sticky</td>
<td>(</td>
<td>S</td>
</tr>
</tbody>
</table>

- **Linear**: the rewriting step replaces an atom with one atom
- **Acyclic**: the rewriting can be seen as a tree of depth at most \(#\text{pred}(O)\)
- **Sticky**: only variables of \( q \) occur more than once in a disjunct
Upper/Lower Bound for **Frontier-Guarded**

- The automata-based approach provides a UCQ-rewriting - disjunction of the trees accepted by the automaton (very large - 5EXP)

- Triple-exponential lower bound for the size of UCQ-rewritings for \((\textbf{EL},\textbf{CQ})\)

[Bienvenu, Lutz & Wolter, IJCAI 2013]
Target More Succinct Query Languages

In particular, what about

- Positive existential queries (PE)
- Non-recursive Datalog queries (NDL)
- First-order queries (FO)

Even for \((\text{DL-Lite}_R, \text{CQ})\)

- No PE/NDL-rewriting of polynomial size
- No FO-rewriting of polynomial size (unless the PH collapses)

…it holds even for \((\text{Acyclic}, \text{CQ})\)

FO-Rewritability: Pure Approach

Two crucial limitations:

- No small rewritings - even for lightweight languages like Linear or DL-Lite$^R$

- Simple OMQs are immediately excluded, e.g.,

$$\langle \{\text{HasChild}, \text{Human}\}, \{\text{HasChild}(x,y), \text{Human}(y) \rightarrow \text{Human}(x)\}, \text{Human}(x) \rangle$$

a more refined approach is needed
FO-Rewritability: Combined Approach

\[ Q = (S, O, q(x_1, \ldots, x_n)) \]

\[ Q_{\text{rew}}(x_1, \ldots, x_n) \]

both steps in polynomial time!!!

for every S-database \( D \): \( Q(D) = Q_{\text{rew}}(D_O) \)

[Lutz, Toman & Wolter, IJCAI 2009]
FO-Rewritability: Combined Approach

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>✓</td>
<td>[×]</td>
<td>[[×]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>[×]</td>
<td>✓</td>
<td>[[×]]</td>
<td>×</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\infty$</td>
<td>✓</td>
<td>✓</td>
<td>[[×]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

[×] - assuming PSPACE $\neq$ NEXPTIME
[[×]] - assuming PSPACE $\neq$ EXPTIME
### FO-Rewritability: Combined Approach

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>✓</td>
<td>[✗]</td>
<td>[[✗]]</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>[✗]</td>
<td>✓</td>
<td>[[✗]]</td>
<td>✗</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\infty$</td>
<td>✓</td>
<td>✓</td>
<td>[[✗]]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>

**via the Polynomial Witness Property**

Definition: \((L, CQ)\) enjoys the PWP if there exists a polynomial \(\text{pol}(\cdot)\) such that for every \(Q = (S, O, q(x)) \in (L, CQ)\), \(S\)-database \(D\), and \(t \in \text{dom}(D)^{|x|}\):

\[ t \in Q(D) \Rightarrow q(t) \text{ can be entailed after } \text{pol}(|O|,|q|) \text{ chase steps} \]
Polynomial Witness Property (PWP)

**Proposition:** PWP $\Rightarrow$ PE/NDL-rewritings constructible in polynomial time, assuming databases with at least two constants

\[ \text{pol}(|O|,|q|) \text{ chase steps} \]
## FO-Rewritability: Combined Approach

### Schema Assumptions

<table>
<thead>
<tr>
<th>Size</th>
<th>Aritiy</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>✓</td>
<td>[✗]</td>
<td>[[✗]]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>[✗]</td>
<td>✓</td>
<td>[[✗]]</td>
<td>✗</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\infty$</td>
<td>✓</td>
<td>✓</td>
<td>[[✗]]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

*via the Polynomial Witness Property*

### FO-Rewritability: Combined Approach

#### Schema Assumptions

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>∞</td>
<td>∞</td>
<td>✓</td>
<td>[✗]</td>
<td>[[✗]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>∞</td>
<td>≤ k</td>
<td>✓</td>
<td>[✗]</td>
<td>✓</td>
<td>[✗]</td>
<td>×</td>
</tr>
<tr>
<td>≤ k</td>
<td>∞</td>
<td>✓</td>
<td>✓</td>
<td>[[✗]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>≤ k</td>
<td>≤ k</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

**via the Polynomial Witness Property?**

FO-Rewritability: Combined Approach

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>✓</td>
<td>[✗]</td>
<td>[[✗]]</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>[✗]</td>
<td>✓</td>
<td>[[✗]]</td>
<td>✓</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\infty$</td>
<td>✓</td>
<td>✓</td>
<td>[[✗]]</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

**via proof generators**

a compact representation of an exponentially-sized witness

[Gottlob, Manna & P., IJCAI 2015]
Proof Generator

$q = \exists x \exists y \exists z \exists w \ P(x,a,y) \land P(z,y,b) \land P(w,c,b)$
Proof Generator

\[ k = (|q| + 1) \cdot (2 \cdot \text{arity})^{\text{arity}} \]

\[ q = \exists x \exists y \exists z \exists w \ P(x,a,y) \land P(z,y,b) \land P(w,c,b) \]

\[ \alpha = (\ldots z_1 \ldots) \]
\[ \beta = (\ldots z_2 \ldots) \]
\[ \gamma = (\ldots z_4 \ldots) \]
\[ \delta = (\ldots z_3 \ldots) \]

chase forest

check via a FO/NDL query whether a proof generator exists

D

P(z_2,a,z_1)

P(z_3,z_1,b)

P(b,z_4,c)
## FO-Rewritability: Combined Approach

### Schema Assumptions

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>✓</td>
<td>[×]</td>
<td>[[×]]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>[×]</td>
<td>✓</td>
<td>[[×]]</td>
<td>✗</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\infty$</td>
<td>✓</td>
<td>✓</td>
<td>[[×]]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>$\leq k$</td>
<td>$\leq k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

A unique positive case without polynomially-sized witnesses

[Gottlob, Manna & P., IJCAI 2015]
### FO-Rewritability: Combined Approach

The table summarizes schema assumptions via linearization:

- **Encode the type of the guard-atom in a single predicate**

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>∞</td>
<td>∞</td>
<td>✓</td>
<td>[×]</td>
<td>[[×]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>∞</td>
<td>≤ k</td>
<td>✓</td>
<td>[×]</td>
<td>✓</td>
<td>[[×]]</td>
<td>×</td>
</tr>
<tr>
<td>≤ k</td>
<td>∞</td>
<td>✓</td>
<td>✓</td>
<td>[[×]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>≤ k</td>
<td>≤ k</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

FO-Rewritability: Combined Approach

<table>
<thead>
<tr>
<th>Size</th>
<th>Arity</th>
<th>Linear</th>
<th>Acyclic</th>
<th>Sticky</th>
<th>Guarded</th>
<th>Fr-Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>✓</td>
<td>[×]</td>
<td>[[×]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$\infty$</td>
<td>≤ $k$</td>
<td>✓</td>
<td>[×]</td>
<td>✓</td>
<td>[[×]]</td>
<td>×</td>
</tr>
<tr>
<td>≤ $k$</td>
<td>$\infty$</td>
<td>✓</td>
<td>✓</td>
<td>[[×]]</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>≤ $k$</td>
<td>≤ $k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
</tbody>
</table>

Fixing the schema is not enough.
We should fix the ontology, and then adapt the linearization technique.

[Thomazo, Personal Communication 2017]
Some Final Remarks

- **FO-Rewritable languages**
  - Practical resolution-based algorithms exist (XRewrite, Piece-based rewriting)
  - Prototype systems exist (Nyaya, Graal)

- **Far from practical algorithms for checking FO rewritability**
  - Notable exception the algorithm for (EL, CQ)
  - Prototype system Grind

- **Polynomial combined FO rewriting algorithms are of theoretical nature**
  - Can we construct compact UCQs?
Thank you!