Chasing Sets: How to Use Existential Rules for Expressive Reasoning

David Carral, Irina Dragoste, Markus Krötzsch, Christian Lewe
Datalog for DL reasoning?

Can we use Datalog to solve hard problems?
  • ExpTime-complete combined complexity
  • Fast and scalable reasoners available
Datalog for DL reasoning?

Can we use Datalog to solve hard problems?
  • ExpTime-complete combined complexity
  • Fast and scalable reasoners available

✓ Carral et al. (KR 2018): Horn-ALCHOIQ
Datalog for DL reasoning?

Can we use Datalog to solve hard problems?
  • ExpTime-complete combined complexity
  • Fast and scalable reasoners available

✓ Carral et al. (KR 2018): Horn-ALCHOIQ
  ➡ exponentially many rules
Datalog for DL reasoning?

Can we use Datalog to solve hard problems?

- ExpTime-complete combined complexity
- Fast and scalable reasoners available

✓ Carral et al. (KR 2018): Horn-ALCHOIQ
  ➡ exponentially many rules

✓ Ahmetaj et al. (ICDT 2018): guarded existential rules
Datalog for DL reasoning?

Can we use Datalog to solve hard problems?
- ExpTime-complete combined complexity
- Fast and scalable reasoners available

✓ Carral et al. (KR 2018): Horn-ALCHOIQ
  ➡ exponentially many rules

✓ Ahmetaj et al. (ICDT 2018): guarded existential rules
  ➡ predicates with linearly large arities
Is there an efficient way to solve hard problems with rule engines, nonetheless?
Our contribution
Our contribution

By moving from **Datalog** to **existential rules** we can
Our contribution

By moving from Datalog to existential rules we can
  • solve hard (ExpTime-complete) real-world problems
Our contribution

By moving from Datalog to existential rules we can
  • solve hard (ExpTime-complete) real-world problems
  • using existing rule engines
Our contribution

By moving from Datalog to existential rules we can

• solve hard (ExpTime-complete) real-world problems
• using existing rule engines
• with a fixed set of rules
Our contribution

By moving from **Datalog** to **existential rules** we can

- solve hard (ExpTime-complete) real-world problems
- using existing rule engines
- with a fixed set of rules
Our contribution

By moving from Datalog to existential rules we can

• solve hard (ExpTime-complete) real-world problems
• using existing rule engines
• with a fixed set of rules

› Available reasoners use the chase algorithm
Our contribution

By moving from **Datalog** to **existential rules** we can

- solve hard (ExpTime-complete) real-world problems
- using existing rule engines
- with a fixed set of rules

- Available reasoners use the chase algorithm
- Chase algorithm may not terminate
Our contribution

By moving from **Datalog** to **existential rules** we can

- solve hard (ExpTime-complete) real-world problems
- using existing rule engines
- with a fixed set of rules

- Available reasoners use the **chase algorithm**
- Chase algorithm **may not terminate**
- Sufficient conditions for chase termination
Our contribution

By moving from **Datalog** to **existential rules** we can

- solve hard (ExpTime-complete) real-world problems
- using existing rule engines
- with a fixed set of rules

- Available reasoners use the **chase algorithm**
- Chase algorithm **may not terminate**
- Sufficient conditions for chase termination
  - characterise rule sets of **PTime** data complexity (like Datalog)
How can we get the required expressivity?
Datalog(S)

**Surface language for existential rules with terminating chase**

- ExpTime-complete data complexity
- polynomial translation from Datalog(S) to existential rules
Datalog(S)

**Surface language for existential rules with terminating chase**

- ExpTime-complete data complexity
- Polynomial translation from Datalog(S) to existential rules

\[
\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \tag{1}
\]

\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \tag{2}
\]
Datalog(S)

Surface language for existential rules with terminating chase

- ExpTime-complete data complexity
- Polynomial translation from Datalog(S) to existential rules

\[
\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset)
\]  

\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\})
\]  

\[
\text{likesAll}(x, S) \rightarrow \text{allLikeAll}({x}, S)
\]  

\[
\text{allLikeAll}(S, T) \land \text{likesAll}(x, T) \rightarrow \text{allLikeAll}(S \cup \{x\}, T)
\]
Datalog(S)

**Surface language for existential rules with terminating chase**

- ExpTime-complete data complexity
- Polynomial translation from Datalog(S) to existential rules

\[
\begin{align*}
\text{person}(x) &\rightarrow \text{likesAll}(x, \emptyset) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) &\rightarrow \text{likesAll}(x, S \cup \{y\}) \\
\text{likesAll}(x, S) &\rightarrow \text{allLikeAll}(\{x\}, S) \\
\text{allLikeAll}(S, T) \land \text{likesAll}(x, T) &\rightarrow \text{allLikeAll}(S \cup \{x\}, T) \\
\text{allLikeAll}(S, S) \land \text{alice} \in S &\rightarrow \text{cliqueOfAlice}(S)
\end{align*}
\]
Datalog(S): Definition

Logic with two sorts: objects and sets of objects

- Each predicate position has a sort
- Object and set variables are distinct
- Set terms: $\emptyset$  $\{object\}$  $Set_1 \cup Set_2$
- Built-in predicates (only in body): $object \in Set$  $Set_1 \subseteq Set_2$
Datalog(S): Definition

Logic with two sorts: **objects** and **sets of objects**

- Each predicate position has a sort
- Object and set variables are distinct
- Set terms:
  - Built-in predicates (only in body): $object \in Set$  
  - $\emptyset, \{ object \}, Set_1 \cup Set_2, Set_1 \subseteq Set_2$

All set variables must occur in a regular body atom (not built-in)
Datalog(S): Definition

Logic with two sorts: **objects** and **sets of objects**
- Each predicate position has a sort
- Object and set variables are distinct
- Set terms: \( \emptyset \) \( \{ \text{object} \} \) \( \text{Set}_1 \cup \text{Set}_2 \)
- Built-in predicates (only in body): \( \text{object} \in \text{Set} \) \( \text{Set}_1 \subseteq \text{Set}_2 \)

All set variables must occur in a regular body atom (not built-in)

**Theorem:** Datalog(S) has **ExpTime-complete** combined and **data** complexity.
person(x) → likesAll(x, ∅)  \quad (1)
likesAll(x, S) ∧ likes(x, y) → likesAll(x, S \cup \{y\})  \quad (2)
\[\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1)\]
\[\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2)\]
\[\rightarrow \exists V. \text{empty}(V) \quad (1')\]
\[\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'')\]
person(x) → likesAll(x, ∅)  \hspace{1cm} (1)

likesAll(x, S) \land likes(x, y) → likesAll(x, S \cup \{y\})  \hspace{1cm} (2)

\rightarrow \exists V . empty(V)  \hspace{1cm} (1')

person(x) \land empty(Y) → likesAll(x, Y)  \hspace{1cm} (1'')

likesAll(x, S) \land likes(x, y) → \exists V . likesAll(x, V) \land SU(S, y, V)  \hspace{1cm} (2')
\[
\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1)
\]
\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2)
\]
\[
\rightarrow \exists V. \text{empty}(V) \quad (1')
\]
\[
\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'')
\]
\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \quad (2')
\]

\[
\text{person}(Eve)
\]
\[
\text{likes}(Eve, a)
\]
\[
\text{likes}(Eve, b)
\]
\[\begin{align*}
\text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
& \quad \rightarrow \exists V. \text{empty}(V) \quad (1') \\
\text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \exists V. \text{likesAll}(x, V) \land S(U(S, y, V)) \quad (2')
\end{align*}\]

**Eve**

\[\begin{align*}
\text{person}(\text{Eve}) \\
\text{likes}(\text{Eve}, a) \\
\text{likes}(\text{Eve}, b)
\end{align*}\]
\[ \text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \]
\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land S\cup\{y\} \quad (2') \]

\textit{Eve}

\begin{align*}
\text{person(Eve)} \\
\text{likes(Eve, a)} \\
\text{likes(Eve, b)}
\end{align*}

\[ n_{\emptyset} \]
person(x) → likesAll(x, ∅)    (1)
likesAll(x, S) ∧ likes(x, y) → likesAll(x, S ∪ {y})    (2)

→ ∃V. empty(V)          (1')

person(x) ∧ empty(Y) → likesAll(x, Y)      (1'')
likesAll(x, S) ∧ likes(x, y) → ∃V. likesAll(x, V) ∧ SU(S, y, V)    (2')
\[ \text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \]
\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \quad (2') \]
person(x) → likesAll(x, ∅) \quad (1)
likesAll(x, S) ∧ likes(x, y) → likesAll(x, S ∪ \{y\}) \quad (2)

→ ∃V. empty(V) \quad (1')

person(x) ∧ empty(Y) → likesAll(x, Y) \quad (1'')
likesAll(x, S) ∧ likes(x, y) → ∃V. likesAll(x, V) ∧ SU(S, y, V) \quad (2')
\[ \text{person}(x) \to \text{likesAll}(x, \emptyset) \quad (1) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \to \text{likesAll}(x, S \cup \{ y \}) \quad (2) \]
\[ \to \exists V. \text{empty}(V) \quad (1') \]
\[ \text{person}(x) \land \text{empty}(Y) \to \text{likesAll}(x, Y) \quad (1'') \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \to \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \quad (2') \]
person(x) → likesAll(x, ∅)  

likesAll(x, S) ∧ likes(x, y) → likesAll(x, S ∪ {y})  

→ ∃V. empty(V)  

person(x) ∧ empty(Y) → likesAll(x, Y)  

likesAll(x, S) ∧ likes(x, y) → ∃V. likesAll(x, V) ∧ SU(S, y, V)  

\( n_{\{a,a,a\}} \)  

\( n_{\{a,a\}} \)  

\( n_{\{a\}} \)  

\( n_{\emptyset} \)  

Eve

\( person(Eve) \)  

\( likes(Eve, a) \)  

\( likes(Eve, b) \)  

\( \rightarrow \) likesAll  

\( \rightarrow \) SU
\[\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1)\]
\[\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2)\]
\[\rightarrow \exists V. \text{empty}(V) \quad (1')\]
\[\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'')\]
\[\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \quad (2')\]
\[
\begin{align*}
  \text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
  \text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
  & \rightarrow \exists V. \text{empty}(V) \quad (1') \\
  \text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
  \text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2')
\end{align*}
\]
\begin{align*}
&\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
&\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
&\quad \rightarrow \exists V. \text{empty}(V) \quad (1') \\
&\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
&\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2')
\end{align*}

![Diagram showing relationships between entities and properties such as person, likesAll, SU, and Eve. The diagram illustrates how existential rules can be used to express reasoning about sets and relationships.](image-url)
\[ \text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \]
\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2') \]
\[ \text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \]
\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2') \]
\( \text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \)
\( \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \)
\( \rightarrow \exists V. \text{empty}(V) \quad (1') \)
\( \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \)
\( \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2') \)
\( \text{SU}(U, x, V) \land \text{SU}(U, y, U) \rightarrow \text{SU}(V, y, V) \quad (2'') \)
\begin{align*}
\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) & \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) & \quad (2) \\
\rightarrow \exists V. \text{empty}(V) & \quad (1') \\
\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) & \quad (1'') \\
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) & \quad (2') \\
\text{SU}(U, x, V) \land \text{SU}(U, y, U) \rightarrow \text{SU}(V, y, V) & \quad (2'')
\end{align*}
Datalog(S) to existential rules

**Theorem:** Any Datalog(S) rule set can be
- polynomially translated
- into a consequence-preserving set of existential rules
- with a terminating Datalog-first standard chase.

✓ **Datalog-first** is implemented by some rule engines
Datalog(S) for DL Reasoning?
DL Reasoning using Datalog(S)
Classification for Horn-SHIQ  
(Kazakov, IJCAI 2009)

Proof. By applying structural transformation to $O$, we obtain an ontology $O'$ containing only concept inclusions of the form $A_1 \sqsubseteq A_2$, $A \sqsubseteq \text{st}(C_a)$, and $\text{st}(C_c) \sqsubseteq A$, where $C_a$ occurs positively in $O$ and $C_c$ occurs negatively in $O$. Since $O$ is a Horn $\mathcal{SHIQ}$ ontology, $C_a$ can only be of the form $\top$, $\bot$, $\neg C$, $C \sqcap D$, $\exists R.C$, $\forall R.C$, $\geq n.S.C$, or $\leq 1.S.C$, and $C_c$ can only be of the form $\top$, $\bot$, $A$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, or $\forall R.C$.

Concept inclusions of the form $A \sqsubseteq \text{st}(C_a)$ that are not of form (n1), are transformed to form (n1) as follows:
- $A \sqsubseteq \text{st}(\neg n.S.C) \Rightarrow A \sqsubseteq \exists n.S.B_i$, $1 \leq i \leq n$, $B_i \sqcap B_j \sqsubseteq \bot$, $1 \leq i < j \leq n$, where $B_i$ are fresh atomic concepts.

Concept inclusions of the form $\text{st}(C_a) \sqsubseteq A$ that are not of form (n1) are transformed to form (n1) as follows:
- $\text{st}(C \sqcap D) = A_c \sqcup A_d \sqsubseteq A$ \Rightarrow $A_c \sqsubseteq A$, $A_d \sqsubseteq A$;
- $\text{st}(\exists R.C) = \exists R.A_c \sqsubseteq A$ \Rightarrow $A_c \sqsubseteq \forall R^-.A$;
- $\text{st}(\geq 1.S.C) = \geq 1.S.A_c \sqsubseteq A$ \Rightarrow $A_c \sqsubseteq \forall S^-$.A.

It is easy to show using Proposition 1, that $O' \models \alpha$ iff $O \models \alpha$ for every axiom $\alpha$ containing no new symbols. $\square$

4.2 Elimination of Transitivity
After normalization, we apply a well-known technique, which allows the elimination of transitivity axioms. Transitivity axioms of form (n3) in Lemma 2 can interact only with axioms $M \sqsubseteq A$ and $M \sqsubseteq \exists R^-_2.N_1$. 

Table 3: Saturation Rules for Horn $\mathcal{SHIQ}$ Ontologies
Consequence-driven classification

\[
H \subseteq \exists R . K \quad H \subseteq A
\]

\[
\frac{}{H \subseteq \exists R . (K \cap B) : A \subseteq \forall R . B \in \emptyset}
\]
Consequence-driven classification

\[
\frac{H \sqsubseteq \exists R . K \quad H \sqsubseteq A}{H \sqsubseteq \exists R . (K \cap B)} : A \sqsubseteq \forall R . B \in \emptyset
\]

\[
\text{Exists}(H, r, K) \land \text{SubClass}(H, a) \land \text{ax}_{\sqsubseteq \forall}(a, r, b) \\
\rightarrow \text{Exists}(H, r, K \cup \{b\})
\]
Evaluation

Classification

Class Retrieval

<table>
<thead>
<tr>
<th>Dataset</th>
<th>GO x-anatomy</th>
<th>GO x-taxon</th>
<th>Gazetteer</th>
<th>ChEBI mol. f.c.</th>
<th>NCI</th>
<th>Reactome 1.7M</th>
<th>Reactome 3.1M</th>
<th>Reactome 4.4M</th>
<th>UOBM 1.9M</th>
<th>UOBM 4M</th>
<th>UOBM 5.9M</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>time(s)</strong></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

- VLog
- Konclude
What can we use Datalog(S) for?

Consequence-based classification and class retrieval for Horn-ALC:
• Kazakov (IJCAI 2011)

Fact entailment for guarded existential rules:
• Ahmetaj et al. (ICDT 2018)
What can we use Datalog(S) for?

Consequence-based classification and class retrieval for Horn-ALC:
- Kazakov (IJCAI 2011)

Fact entailment for guarded existential rules:
- Ahmetaj et al. (ICDT 2018)
- Combined-approach materialisation for Horn-ALCHOIQ:
  - Carral et al. (KR 2018)
- Reasoning tasks for non-Horn DLs
  - Simančík et al. (IJCAI 2011): ALCH
What can we use Datalog(S) for?

Consequence-based classification and class retrieval for Horn-ALC:
- Kazakov (IJCAI 2011)

Fact entailment for guarded existential rules:
- Ahmetaj et al. (ICDT 2018)
- Combined-approach materialisation for Horn-ALCHOIQ:
  - Carral et al. (KR 2018)
- Reasoning tasks for non-Horn DLs
  - Simančík et al. (IJCAI 2011): ALCH
- ?
Summary

We provide a practical new way of solving
• ExpTime-complete problems
• using current existential rule engines

Next steps:
• **Logical reasoning**: solve new ExpTime-complete problems
• **Rule engine development**: optimise and benchmark
• **Characterising chase termination**: discover syntactic criteria
JOIN US IN DRESDEN

- Open position available immediately
- Ph.D. students & postdocs
- 100% English & International
- KR / Databases / AI

CONTACT

Markus Krötzsch  markus.kroetzsch@tu-dresden.de