PRACTICAL USES OF EXISTENTIAL RULES IN KNOWLEDGE REPRESENTATION

Part 3: Implementing a Calculus for Horn-\(\mathcal{ALC}\) using Existential Rules

David Carral,\(^1\) Markus Krötzsch,\(^1\) and Jacopo Urbani\(^2\)

1. TU Dresden
2. VU University Amsterdam

Special thanks to Irina Dragoste,\(^1\) Ceriel Jacobs,\(^2\) and Maximilian Marx\(^1\)
for their invaluable contributions to the software used in this tutorial

ECAI, September 4, 2020
**Definition.** A Horn-\(\mathcal{ALC}\) ontology is a set of Horn-\(\mathcal{ALC}\) axioms:

\[
\begin{align*}
A &\subseteq \bot \\
\top &\subseteq B \\
A &\subseteq B \\
A \cap E &\subseteq B \\
\exists R.A &\subseteq B \\
A &\subseteq \forall R.B \\
A &\subseteq \exists R.B
\end{align*}
\]

In the above; \(A\), \(B\), and \(E\) are concept names; and \(R\) is a role name.

**Remark.** Note the axioms of the form \(A \subseteq \forall R.B\), which are not \(\mathcal{EL}\), such as:

\[
\text{CheesePizza} \subseteq \forall \text{HasTopping}.\text{Cheese}
\]

The axiom states that “all toppings in a cheese pizza are cheese toppings”.

Even though Horn-\(\mathcal{ALC}\) is not much more expressive than \(\mathcal{EL}\), (Krötzsch, Rudolph, and Hitzler 2013) have showed that:

**Theorem.** Solving classification over Horn-\(\mathcal{ALC}\) is ExpTime-complete.
**Definition.** We define the semantics of Horn-\(\mathcal{ALC}\) axioms via translation into equivalent first-order logic formulas:

\[
\begin{align*}
A \sqsubseteq \bot & \iff \forall x. (A(x) \rightarrow \bot) \\
\top \sqsubseteq B & \iff \forall x. B(x) \\
A \sqsubseteq B & \iff \forall x. (A(x) \rightarrow B(x)) \\
A \sqcap E \sqsubseteq B & \iff \forall x. (A(x) \land E(x) \rightarrow B(x)) \\
\exists R.A \sqsubseteq B & \iff \forall x. (R(x, y) \land A(y) \rightarrow B(x)) \\
A \sqsubseteq \forall R.B & \iff \forall x. (A(x) \land R(x, y) \rightarrow B(y)) \\
A \sqsubseteq \exists R.B & \iff \forall x. (A(x) \rightarrow \exists y. R(x, y) \land B(y))
\end{align*}
\]

In the above; \(A\), \(B\), and \(E\) are concept names, and \(R\) is a role name.
**Definition.** We define the semantics of Horn-\(\mathcal{ALC}\) axioms via translation into equivalent first-order logic formulas:

\[
\begin{align*}
A \sqsubseteq \bot & \iff \forall x. (A(x) \rightarrow \bot) \\
\top \sqsubseteq B & \iff \forall x. B(x) \\
A \sqsubseteq B & \iff \forall x. (A(x) \rightarrow B(x)) \\
A \sqcap E \sqsubseteq B & \iff \forall x. (A(x) \land E(x) \rightarrow B(x)) \\
\exists R.A \sqsubseteq B & \iff \forall x. (R(x, y) \land A(y) \rightarrow B(x)) \\
A \sqsubseteq \forall R.B & \iff \forall x. (A(x) \land R(x, y) \rightarrow B(y)) \\
A \sqsubseteq \exists R.B & \iff \forall x. (A(x) \rightarrow \exists y. (R(x, y) \land B(y)))
\end{align*}
\]

In the above; \(A\), \(B\), and \(E\) are concept names, and \(R\) is a role name.

Often, we remove universal quantifiers from first-order logic formulas.
A Consequence-Based Calculus to Solve Classification

\[
\begin{align*}
R^C_A & : A \in \text{Concepts}(O) \\
R^3_A & : D \in D \\
R^1_n & : \top \in B \in O \\
R^2_n & : A \sqcap E \in B \in O \\
R^+ & : A \in \exists R.B \in O \\
R^- & : \exists R.A \sqsubseteq B \in O \\
R^1 & : B \sqsubseteq \bot \\
R^2 & : A \in \forall R.B \in O
\end{align*}
\]

Figure: Classification Calculus for Horn-\(\mathcal{ALC}\). Where \(A\), \(B\), and \(E\) are concept names; \(R\) is a role name; and \(C\) and \(D\) are conjunctions of concept names

Remark. Original calculus by (Kazakov 2009).
Consequence-Based Calculus: Soundness

**Soundness.** Show via induction that each rule only produces sound inferences.

For instance, let us show that the following production rule is indeed sound:

\[
(R_y) \quad \frac{C \subseteq \exists R.D \quad C \subseteq A}{C \subseteq \exists R.(D \cap B)} : A \subseteq \forall R.B \in O
\]

**Proof:**

1. By IH: \( O \models \bigwedge_{C \in C} C(x) \rightarrow \exists y.(R(x, y) \land \bigwedge_{D \in D} D(y)) \)
2. By IH: \( O \models \bigwedge_{C \in C} C(x) \rightarrow A(x) \)
3. By the precondition of the rule: \( O \models A(x) \land R(x, y) \rightarrow B(y) \)
4. By (1–3) and the semantics of first-order logic:
   \( O \models \bigwedge_{C \in C} C(x) \rightarrow \exists y.(R(x, y) \land \bigwedge_{D \in D} D(y) \land B(y)) \)
To show **completeness**, we verify the following theorem:

**Theorem.** If an axiom of the form $A \sqsubseteq B$ is not derived by the previously proposed calculus on input $O$, then $O \not\models A \sqsubseteq B$.

**Proof Sketch:** Using the output of the calculus on input $O$, we can construct a model for this ontology that contains an element that is in the domain of $A$ but not in the domain of $B$. Therefore, $O \not\models A \sqsubseteq B$.

**Remark.** For a complete proof, check the following references:

- (Kazakov 2009)
- (Simancik, Kazakov, and Horrocks 2011)
Theorem. The Horn-\(\mathcal{ALC}\) classification calculus runs in exponential time in the size of the input ontology \(O\).

Remark. Note that this calculus produces inferences of the form

\[
(1) \quad C \sqsubseteq B \quad \text{and} \quad (2) \quad C \sqsubseteq \exists R. D
\]

where \(B\) is a concept name, \(R\) is a role name, and \(C\) and \(D\) are conjunctions of concept names. Therefore, the calculus may produce at most

\[
2^{\left|\text{Concepts}(O)\right|} \times \left|\text{Concepts}(O)\right| \quad \text{and} \quad 2 \times 2^{\left|\text{Concepts}(O)\right|} \times \left|\text{Roles}(O)\right|
\]

inferences of type (1) and (2), respectively.
Because of the following result, we can not implement the Horn-\(\mathcal{ALC}\) classification calculus using a fixed Datalog rule set:

**Theorem.** The data complexity of fact entailment over Datalog is in P.

**Proof:**

1. Consider a Datalog rule set \(\mathcal{R}\), a fact set \(\mathcal{F}\), and a fact \(\varphi\).
2. Let \(\mathcal{R}'\) be the grounding of \(\mathcal{R}\) using the constants in \(\mathcal{F}\).
3. We have that \(\mathcal{R}' \cup \mathcal{F} \models \varphi\) if and only if \(\mathcal{R} \cup \mathcal{F} \models \varphi\).
4. Checking if \(\mathcal{R}' \cup \mathcal{F} \models \varphi\) can be reduced to fact entailment over propositional logic, which can be solved in polynomial time.
5. If \(\mathcal{R}\) is fixed, then \(\mathcal{R}'\) is polynomial in the number of constants in \(\mathcal{F}\).
6. By (3) and (4): if \(\mathcal{R}\) is fixed, we can decide if \(\mathcal{R} \cup \mathcal{F} \models \varphi\) in polynomial time.
Implementing the Consequence-Based Calculus: Datalog

Because of the following result, we can not implement the Horn-$\mathcal{ALC}$ classification calculus using a fixed Datalog rule set:

**Theorem.** The data complexity of fact entailment over Datalog is in P.

Assume that we can implement the Horn-$\mathcal{ALC}$ classification calculus with a fixed Datalog rule set (as done with the $\mathcal{EL}$ classification calculus). Then:

1. By Theorem 3.4: we could solve Horn-$\mathcal{ALC}$ classification in polynomial time.
3. By (2): $P = \text{ExpTime}$

**Remark.** To implement the Horn-$\mathcal{ALC}$ classification calculus (or any other procedure that solves Horn-$\mathcal{ALC}$ classification), we need a rule-based language with ExpTime-hard data complexity!
We study Datalog(S), an extension of Datalog that can model exponential computations.

**Example.** Consider the following Datalog(S) rule set:

- \( \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \)
- \( \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \)
- \( \text{LikesAll}(x, X) \rightarrow \text{AllLikeAll}(\{x\}, X) \)
- \( \text{AllLikeAll}(X, Y) \land \text{LikesAll}(x, Y) \rightarrow \text{AllLikeAll}(X \cup \{x\}, Y) \)
- \( \text{AllLikeAll}(X, X) \land \text{alice} \in X \rightarrow \text{CliqueOfAlice}(X) \)

**Theorem.** Checking fact entailment for Datalog(S) is ExpTime-complete for both data and combined complexity.

See (Carral et al. 2019) for a complete proof of the above result.
Using a function to encode the axioms and entities in an input ontology as facts and a fixed Datalog(S) rule set, we can implement the Horn-\(\mathcal{ALC}\) classification calculus.

**Example.** For an ontology \(O\), let Facts\((O)\) be the fact set such that:

\[
\begin{align*}
A \sqsubseteq \bot & \in O \mapsto \text{ax}_{\sqsubseteq}(c_A, c_\bot) \\
\top \sqsubseteq B & \in O \mapsto \text{ax}_{\sqsubseteq}(c_\top, c_B) \\
A \sqsubseteq B & \in O \mapsto \text{ax}_{\sqsubseteq}(c_A, c_B) \\
A \sqcap E \sqsubseteq B & \in O \mapsto \text{ax}_{\sqcap}(c_A, c_E, c_B) \\
\end{align*}
\]

\[
\begin{align*}
\exists R. A \sqsubseteq B & \in O \mapsto \text{ax}_{\exists}(c_A, c_R, c_B) \\
A \sqsubseteq \forall R. B & \in O \mapsto \text{ax}_{\forall}(c_A, c_R, c_B) \\
A \sqsubseteq \exists R. B & \in O \mapsto \text{ax}_{\exists}(c_A, c_R, c_B) \\
A \in \text{Concepts}(O) & \mapsto \text{Concept}(c_E) \\
\end{align*}
\]

In the above; \(c_A, c_B, c_E, c_\top,\) and \(c_\bot\) are fresh constants unique for \(A, B, E, \top,\) and \(\bot,\) respectively; and \(c_R\) is a fresh constant unique \(R.\)
We translate the production rules in the Horn-$\mathcal{ALC}$ classification calculus (left) into analogous Datalog(S) rules (right):

\[
\begin{align*}
R_{A}^{C} & : A \subseteq A & A \in \text{Concepts}(O) & \rightarrow \text{SC}(\{x\}, x) \\
R_{A}^{3} & : C \subseteq \exists R.D & D \subseteq D & \rightarrow \text{SC}(D, d) \\
R_{T}^{0} & : C \subseteq A & B \in O & \rightarrow \text{SC}(C, b) \\
R_{T}^{1} & : C \subseteq A & A \subseteq B \in O & \rightarrow \text{SC}(C, b) \\
R_{T}^{2} & : C \subseteq A & C \subseteq E & A \cap E \subseteq B \in O & \rightarrow \text{SC}(C, b)
\end{align*}
\]
A Horn-$\mathcal{ALC}$ Classification Calculus with Datalog(S)

We translate the production rules in the Horn-$\mathcal{ALC}$ classification calculus (left) into analogous Datalog(S) rules (right):

\[
\begin{align*}
R_+^+ \quad &\quad C \subseteq A \\ &\quad \frac{}{C \subseteq \exists R.B} : A \subseteq \exists R.B \in O \\
R_+^- \quad &\quad C \subseteq \exists R.D \\ &\quad \frac{D \subseteq A}{C \subseteq B} : \exists R.A \subseteq B \in O \\
R_+^\lambda \quad &\quad C \subseteq \exists R.D \\ &\quad \frac{D \subseteq \bot}{C \subseteq \bot} \\
R_+^\chi \quad &\quad C \subseteq \exists R.D \\ &\quad \frac{C \subseteq A}{C \subseteq \exists R.(D \cap B)} : A \subseteq \forall R.B
\end{align*}
\]

\[
\begin{align*}
&\quad SC(C, a) \land ax_{\exists}(a, r, b) \\
&\quad \rightarrow \quad \text{Ex}(C, r, \{B\}) \\
&\quad \text{Ex}(C, r, D) \land SC(D, a) \land ax_{\exists}(r, a, b) \\
&\quad \rightarrow \quad \text{SC}(C, b) \\
&\quad \text{Ex}(C, r, D) \land SC(D, c_{\bot}) \\
&\quad \rightarrow \quad \text{SC}(C, c_{\bot}) \\
&\quad \text{Ex}(C, r, D) \land SC(C, a) \land ax_{\forall}(a, r, b) \\
&\quad \rightarrow \quad \text{Ex}(C, r, D \cup \{b\})
\end{align*}
\]
**Definition.** Let $R_{HALC}$ be the following Datalog(S) rule set:

- $\text{Concept}(x) \rightarrow \text{SC}({x}, x)$
- $\text{Ex}(C, r, D) \land d \in D \rightarrow \text{SC}(D, d)$
- $\text{SC}(C, a) \land \text{ax}_{\sqsubseteq}(c_T, b) \rightarrow \text{SC}(C, b)$
- $\text{SC}(C, a) \land \text{ax}_{\sqsubseteq}(a, b) \rightarrow \text{SC}(C, b)$
- $\text{SC}(C, a) \land \text{SC}(C, e) \land \text{ax}_{\sqsubseteq}(a, e, b) \rightarrow \text{SC}(C, b)$
- $\text{SC}(C, a) \land \text{ax}_{\exists}(a, r, b) \rightarrow \text{Ex}(C, r, \{B\})$
- $\text{Ex}(C, r, D) \land \text{SC}(D, a) \land \text{ax}_{\exists}(r, a, b) \rightarrow \text{SC}(C, b)$
- $\text{Ex}(C, r, D) \land \text{SC}(D, c_\bot) \rightarrow \text{SC}(C, c_\bot)$
- $\text{Ex}(C, r, D) \land \text{SC}(C, a) \land \text{ax}_{\forall}(a, r, b) \rightarrow \text{Ex}(C, r, D \cup \{b\})$

**Theorem.** Consider a Horn-$\mathcal{ALC}$ ontology $O$ and an axiom of the form $A \sqsubseteq B$. Then, $O \models A \sqsubseteq B$ if and only if $R_{HALC} \cup \text{Facts}(O) \models \text{SC}(c_A, c_B)$. 
Alas, VLog does not support Datalog(S) reasoning. There maybe some other rule-based language that we can use...

The following result is a recent finding by (Krötzsch, Marx, and Rudolph 2019):

**Theorem.** The data complexity of fact entailment over rule sets that terminate with respect to the restricted chase is ExpTime-hard.

Moreover, (Carral et al. 2019) have proposed a translation from Datalog(S) into existential rule programs such that:

- The resulting programs terminate with respect to the restricted chase.
- Fact entailment is preserved.
From Datalog(S) to Existential Rules

\[
\begin{align*}
\text{Person}(x) & \rightarrow \text{LikesAll}(x, \emptyset) & \text{LikesAll}(x, X) \land \text{Likes}(x, y) & \rightarrow \text{LikesAll}(x, X \cup \{y\})
\end{align*}
\]

\[
\begin{align*}
\rightarrow \exists V. \text{empty}(V) & \quad (1.1) \\
\text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) & \quad (1.2)
\end{align*}
\]
### From Datalog\(S\) to Existential Rules

<table>
<thead>
<tr>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person(x) → LikesAll(x, \emptyset)</td>
</tr>
<tr>
<td>LikesAll(x, X) ∧ Likes(x, y) → LikesAll(x, X \cup {y})</td>
</tr>
</tbody>
</table>

\[(1.1) \quad \forall V. \text{empty}(V)\]

\[(1.2) \quad \text{person}(x) ∧ \text{empty}(Y) \rightarrow \text{likesAll}(x, Y)\]

\[(2.1) \quad \text{likesAll}(x, S) ∧ \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) ∧ \text{SU}(S, y, V)\]
From Datalog(S) to Existential Rules

Person(x) → LikesAll(x, ∅)  
LikesAll(x, X) ∧ Likes(x, y) → LikesAll(x, X ∪ {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)

person(eve)  
likes(eve, a)  
likes(eve, b)
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \]
\[ \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \quad (1.1) \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1.2) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \quad (2.1) \]

\[ \text{eve} \]

\[ \text{person(eve)} \]
\[ \text{likes(eve, a)} \]
\[ \text{likes(eve, b)} \]
From Datalog(S) to Existential Rules

Person(x) → LikesAll(x, \emptyset)  LikesAll(x, X) ∧ Likes(x, y) → LikesAll(x, X ∪ \{y\})

⇒ ∃V. \text{empty}(V)  \quad \text{(1.1)}

person(x) ∧ \text{empty}(Y) → likesAll(x, Y)  \quad \text{(1.2)}

likesAll(x, S) ∧ \text{likes}(x, y) → ∃V. \text{likesAll}(x, V) ∧ \text{SU}(S, y, V)  \quad \text{(2.1)}

\text{eve}

person(eve)
likes(eve, a)
likes(eve, b)

n\emptyset
From Datalog(S) to Existential Rules

\[
\text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \quad \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\})
\]

\[
\rightarrow \exists V. \text{empty}(V) \quad (1.1)
\]

\[
\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1.2)
\]

\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land S(U, y, V) \quad (2.1)
\]
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \quad \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \quad \text{(1.1)} \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad \text{(1.2)} \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \quad \text{(2.1)} \]
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \]

\[ \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \]  \hspace{2cm} (1.1)

\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \]  \hspace{2cm} (1.2)

\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \]  \hspace{2cm} (2.1)

Diagram:
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \]

\[ \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \]

\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \tag{1.2} \]

\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \tag{2.1} \]
From Datalog(S) to Existential Rules

Person(x) → LikesAll(x, {})
LikesAll(x, X) ∧ Likes(x, y) → LikesAll(x, X ∪ {y})

→ ∃V. empty(V)  \hspace{2cm} (1.1)

person(x) ∧ empty(Y) → likesAll(x, Y)  \hspace{2cm} (1.2)
likesAll(x, S) ∧ likes(x, y) → ∃V. likesAll(x, V) ∧ SU(S, y, V)  \hspace{2cm} (2.1)
From Datalog(S) to Existential Rules

\[
\text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \quad \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\})
\]

\[
\rightarrow \exists V. \text{empty}(V) \quad (1.1)
\]

\[
\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1.2)
\]

\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \quad (2.1)
\]
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \quad \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \quad (1.1) \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1.2) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \land SU(V, y, V) \quad (2.1) \]

David Carral, September 4, 2020
Practical Uses of Existential Rules in Knowledge Representation
From Datalog(S) to Existential Rules

Person(x) → LikesAll(x, ∅)  
LikesAll(x, X) ∧ Likes(x, y) → LikesAll(x, X ∪ {y})

→ ∃V . empty(V)  
(1.1)

person(x) ∧ empty(Y) → likesAll(x, Y)  
(1.2)

likesAll(x, S) ∧ likes(x, y) → ∃V . likesAll(x, V) ∧ SU(S, y, V) ∧ SU(V, y, V)  
(2.1)

David Carral, September 4, 2020  
Practical Uses of Existential Rules in Knowledge Representation  
slide 18 of 23
Person\( (x) \rightarrow \text{LikesAll}(x, \emptyset) \) \hspace{1cm} \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\})

\[ \rightarrow \exists V. \text{empty}(V) \]  
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \]  
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \]  

\text{Commentary on diagram:}

- Node labeled \( n_{\{a,b,a\}} \) connected to nodes labeled \( a \), \( n_{\{a\}} \), and \( n_{\emptyset} \).
- Edge labeled \( \text{likesAll} \) connects \( n_{\{a\}} \) to \( n_{\emptyset} \).
- Edge labeled \( \text{SU} \) connects \( n_{\emptyset} \) to \( n_{\{b\}} \).
- Node labeled \( \text{person}(\text{eve}) \) with edges labeled \( \text{likes}(\text{eve}, a) \) and \( \text{likes}(\text{eve}, b) \).
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \]
\[ \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \quad \text{(1.1)} \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad \text{(1.2)} \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \land SU(V, y, V) \quad \text{(2.1)} \]
From Datalog(S) to Existential Rules

\[ \text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \quad \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\}) \]

\[ \rightarrow \exists V. \text{empty}(V) \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \] (1.1)
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \] (2.1)
\[ \text{SU}(U, x, V) \land \text{SU}(U, y, U) \rightarrow \text{SU}(V, y, V) \] (2.2)

David Carral, September 4, 2020
Practical Uses of Existential Rules in Knowledge Representation
slide 18 of 23
From Datalog(S) to Existential Rules

\[
\text{Person}(x) \rightarrow \text{LikesAll}(x, \emptyset) \quad \text{LikesAll}(x, X) \land \text{Likes}(x, y) \rightarrow \text{LikesAll}(x, X \cup \{y\})
\]

\[
\rightarrow \exists V. \text{empty}(V)
\]

\[
\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y)
\]

\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V)
\]

\[
\text{SU}(U, x, V) \land \text{SU}(U, y, U) \rightarrow \text{SU}(V, y, V)
\]

David Carral, September 4, 2020
### Experimental Evaluation: Solving Classification

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>#Ax.</td>
<td>#Set</td>
<td>#SC</td>
<td>#Ex</td>
<td>VLog</td>
<td>Konclude</td>
</tr>
<tr>
<td>00040</td>
<td>223K</td>
<td>2K</td>
<td>1051K</td>
<td>334K</td>
<td>432s</td>
<td>5s</td>
</tr>
<tr>
<td>00048</td>
<td>142K</td>
<td>19</td>
<td>718K</td>
<td>171K</td>
<td>387s</td>
<td>3s</td>
</tr>
<tr>
<td>00477</td>
<td>318K</td>
<td>0</td>
<td>162K</td>
<td>167K</td>
<td>1s</td>
<td>3s</td>
</tr>
<tr>
<td>00533</td>
<td>159K</td>
<td>0</td>
<td>965K</td>
<td>351K</td>
<td>132s</td>
<td>2s</td>
</tr>
<tr>
<td>00786</td>
<td>152K</td>
<td>12K</td>
<td>2283K</td>
<td>978K</td>
<td>549s</td>
<td>14s</td>
</tr>
</tbody>
</table>

**Figure:** Ontologies and results for classification (A) showing: axiom count; number of non-singleton “set terms” introduced (#Set); number of SC and Ex facts derived; reasoning time in VLog and Konclude.
**Definition.** A Horn-$\mathcal{ALC}$ ontology is a set of Horn-$\mathcal{ALC}$ axioms:

\[
\begin{align*}
A & \sqsubseteq \bot \\
\top & \sqsubseteq B \\
A & \sqsubseteq B \\
A \cap E & \sqsubseteq B \\
\exists R.A & \sqsubseteq B \\
A & \sqsubseteq \forall R.B \\
A & \sqsubseteq \exists R.B \\
A(a) & \quad R(a, b)
\end{align*}
\]

In the above; $A$, $B$, and $E$ are concept names (i.e., unary predicates); and $R$ is a role name (i.e., binary predicate).

**Definition.** Assertion Retrieval is the reasoning task of computing all axioms of the form $A(a)$ or $R(a, b)$ that are logically entailed by some input ontology $O$.

**Remark.** The Horn-$\mathcal{ALC}$ classification calculus can be extended with 3 rules (as done by (Carral et al. 2019)) to solve assertion retrieval.
Experimental Evaluation: Assertion Retrieval

Figure: Experimental results for class retrieval (B) in VLog (pink/grey) and Konclude (black); note the log scale.
Conclusions and Future Work

**Remark.** We can use VLog to solve ExpTime-hard problems!

Future work:

- Rulewerk Extension: translate Datalog(S) to existential rules
- VLog Extension: native support for Datalog(S)
- Implement existing calculi using our approach

Hands on Session!


