Chasing Sets: How to Use Existential Rules for Expressive Reasoning

David Carral, **Irina Dragoste**, Markus Krötzsch, Christian Lewe

TECHNISCHE UNIVERSITÄT DRESDEN

Datalog for DL reasoning?

Can we use Datalog to solve hard problems?

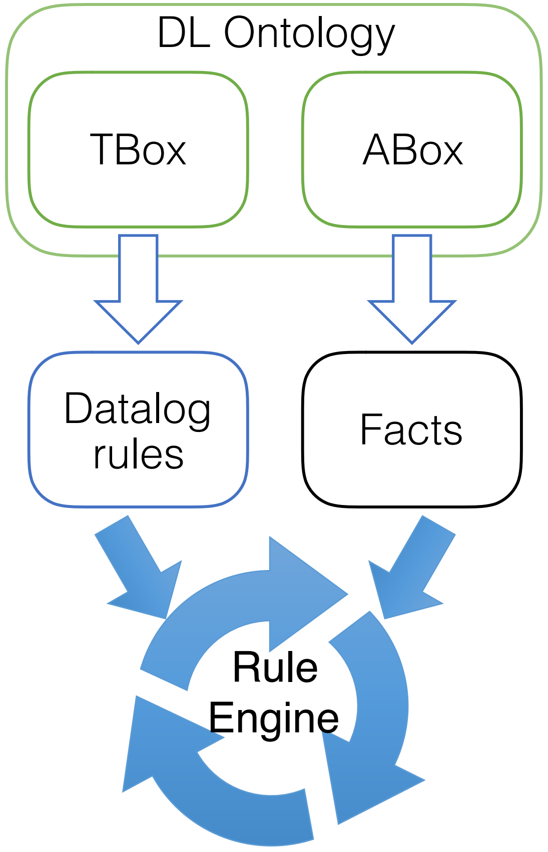
- ExpTime-complete combined complexity
- Fast and scalable reasoners available

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



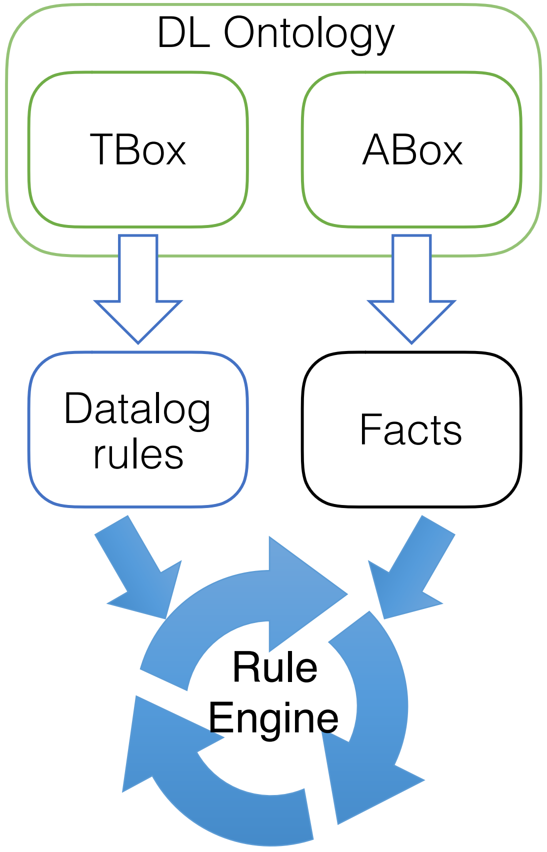
Datalog for DL reasoning?

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?

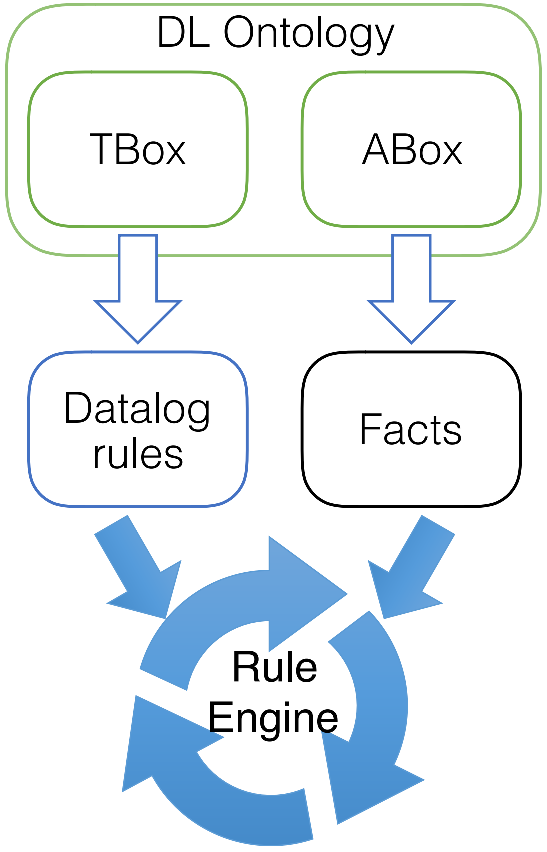
Can we use Datalog to solve hard problems?

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

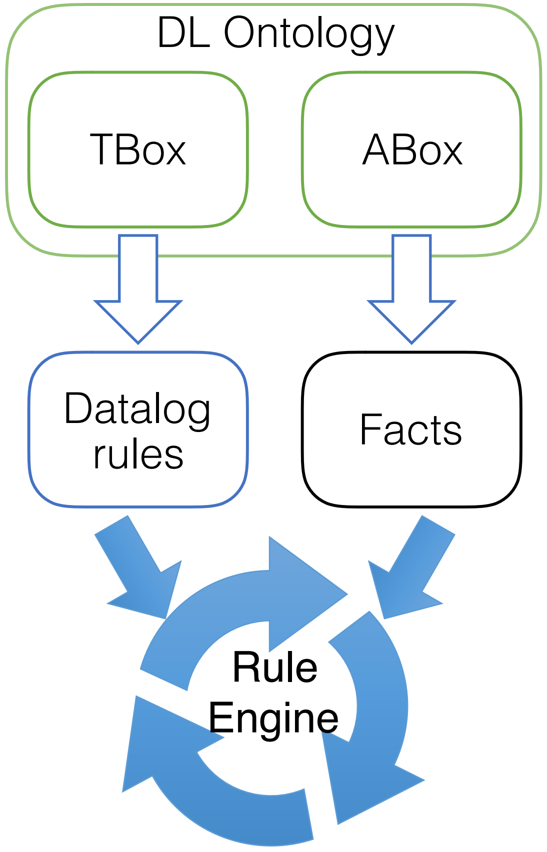
✓Carral et al. (KR 2018): Horn-ALCHOIQ

➡ 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

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

Datalog(S)

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

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

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$likesAll(x, S) \rightarrow allLikeAll(\{x\}, S) \quad (3)$$

$$allLikeAll(S, T) \wedge likesAll(x, T) \rightarrow allLikeAll(S \cup \{x\}, T) \quad (4)$$

Datalog(S)

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

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

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$likesAll(x, S) \rightarrow allLikeAll(\{x\}, S) \quad (3)$$

$$allLikeAll(S, T) \wedge likesAll(x, T) \rightarrow allLikeAll(S \cup \{x\}, T) \quad (4)$$

$$allLikeAll(S, S) \wedge alice \in S \rightarrow cliqueOfAlice(S) \quad (5)$$

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: $\emptyset$ $\{object\}$ $Set_1 \cup Set_2$
- Built-in predicates (only in body): $object \in Set$ $Set_1 \subseteq Set_2$

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: $\emptyset$ $\{object\}$ $Set_1 \cup Set_2$
- Built-in predicates (only in body): $object \in Set$ $Set_1 \subseteq Set_2$

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

Theorem: Datalog(S) has **ExpTime-complete** combined and **data** complexity.

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V . empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$person(x) \to likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \to likesAll(x, S \cup \{y\}) \quad (2)$$

$$\to \exists V \,.\, empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \to likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \to \exists V \,.\, likesAll(x, V) \wedge SU(S, y, V) \quad (2')$$

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V \,.\, empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V \,.\, likesAll(x, V) \wedge SU(S, y, V) \quad (2')$$

$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V \,.\, empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V \,.\, likesAll(x, V) \wedge SU(S, y, V) \quad (2')$$

Eve

$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

$$person(x) \rightarrow likesAll(x, \emptyset) \qquad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \qquad (2)$$

$$\rightarrow \exists V . empty(V) \qquad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \qquad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V . likesAll(x, V) \wedge SU(S, y, V) \qquad (2')$$

Eve

$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

$n_{\emptyset}$

$$person(x) \rightarrow likesAll(x, \varnothing) \quad (1)$$

$$likesAll(x, S) \land likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V . empty(V) \quad (1')$$

$$person(x) \land empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \land likes(x, y) \rightarrow \exists V . likesAll(x, V) \land SU(S, y, V) \quad (2')$$

$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

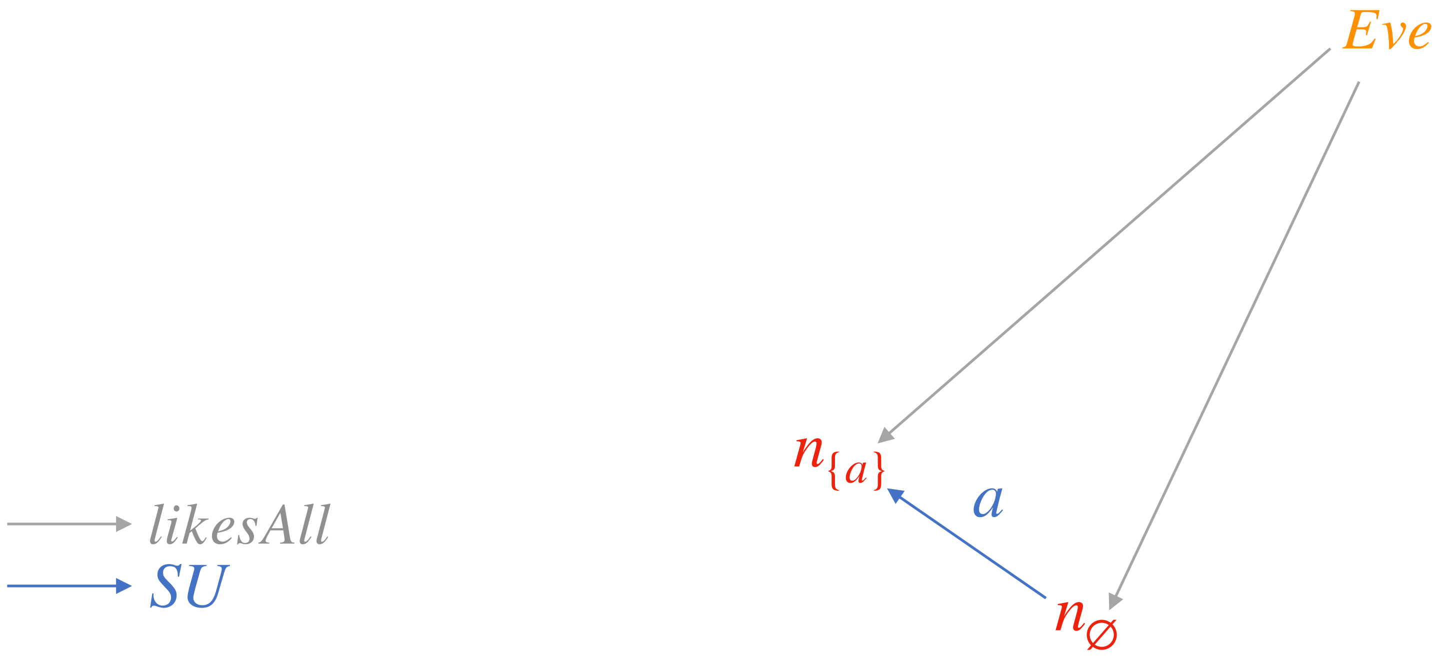
$$person(x) \to likesAll(x, \emptyset) \qquad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \to likesAll(x, S \cup \{y\}) \qquad (2)$$

$$\to \exists V . empty(V) \qquad (1')$$

$$person(x) \wedge empty(Y) \to likesAll(x, Y) \qquad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \to \exists V . likesAll(x, V) \wedge SU(S, y, V) \qquad (2')$$



$person(Eve)$
$likes(Eve, a)$
$likes(Eve, b)$

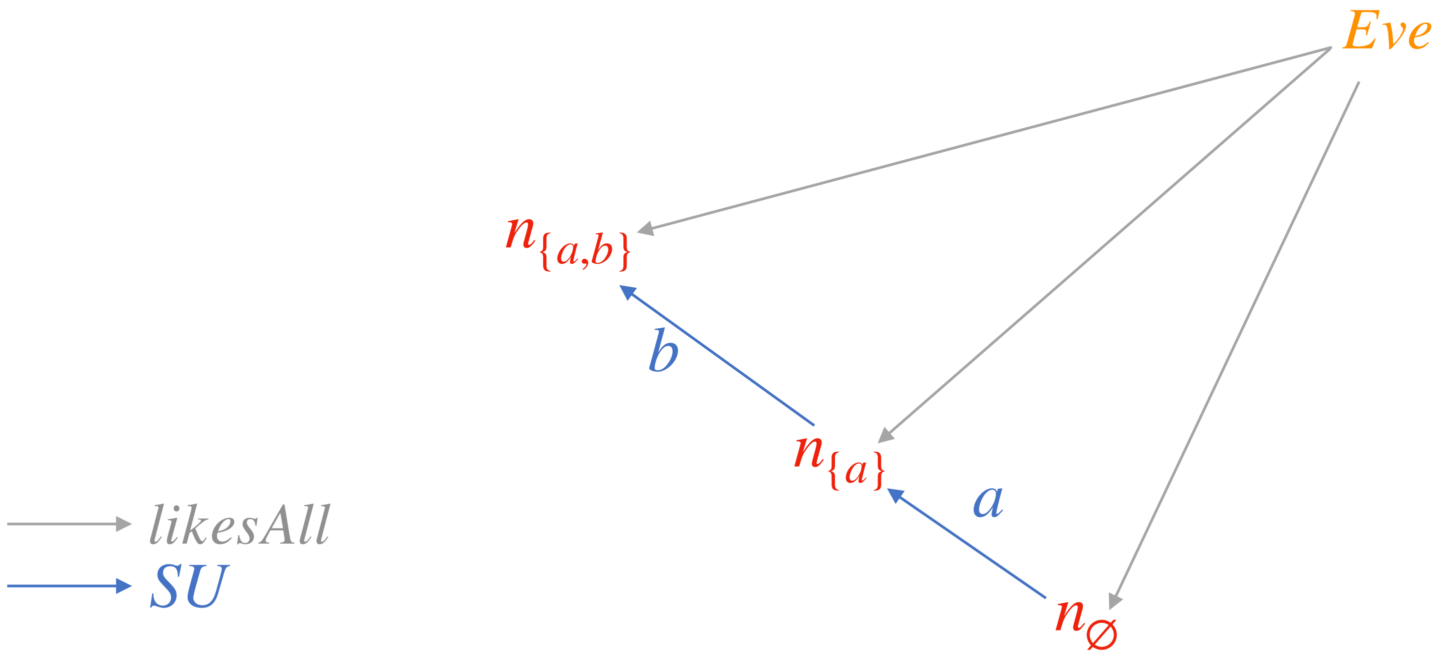
$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V . empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V . likesAll(x, V) \wedge SU(S, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

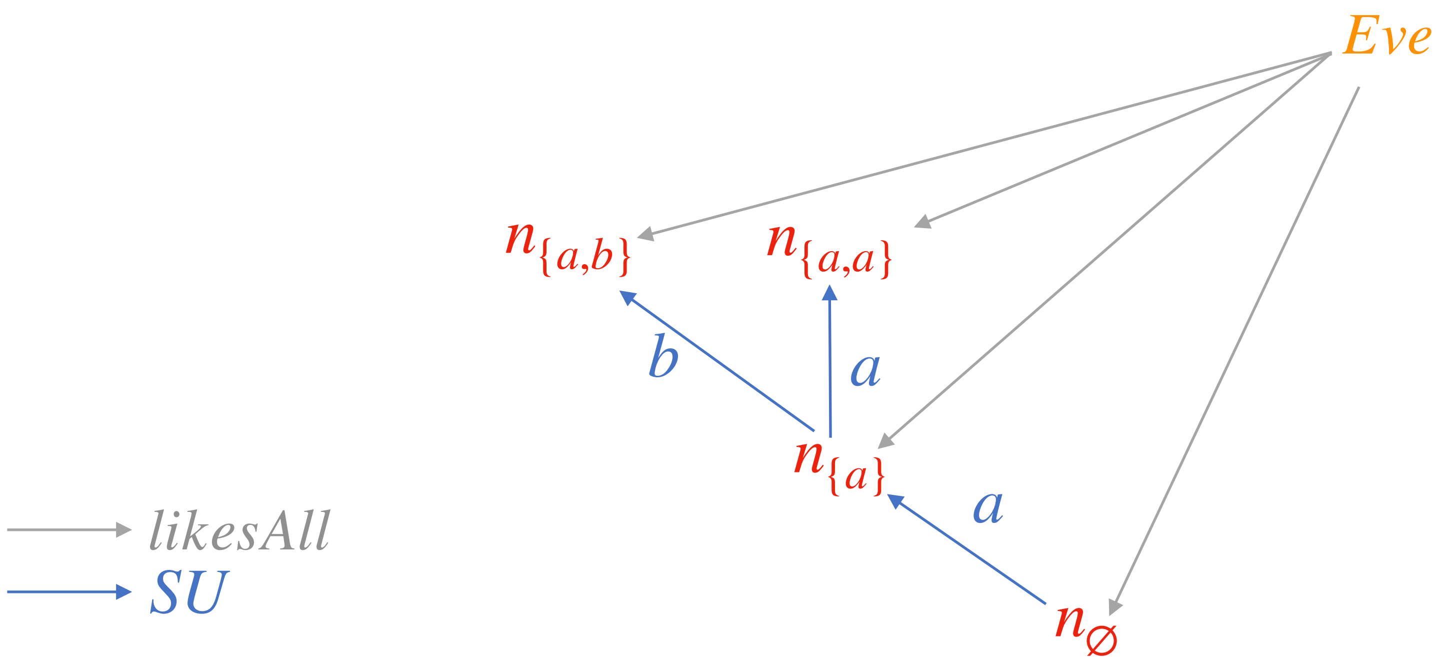
$$person(x) \to likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \to likesAll(x, S \cup \{y\}) \quad (2)$$

$$\to \exists V . empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \to likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \to \exists V . likesAll(x, V) \wedge SU(S, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

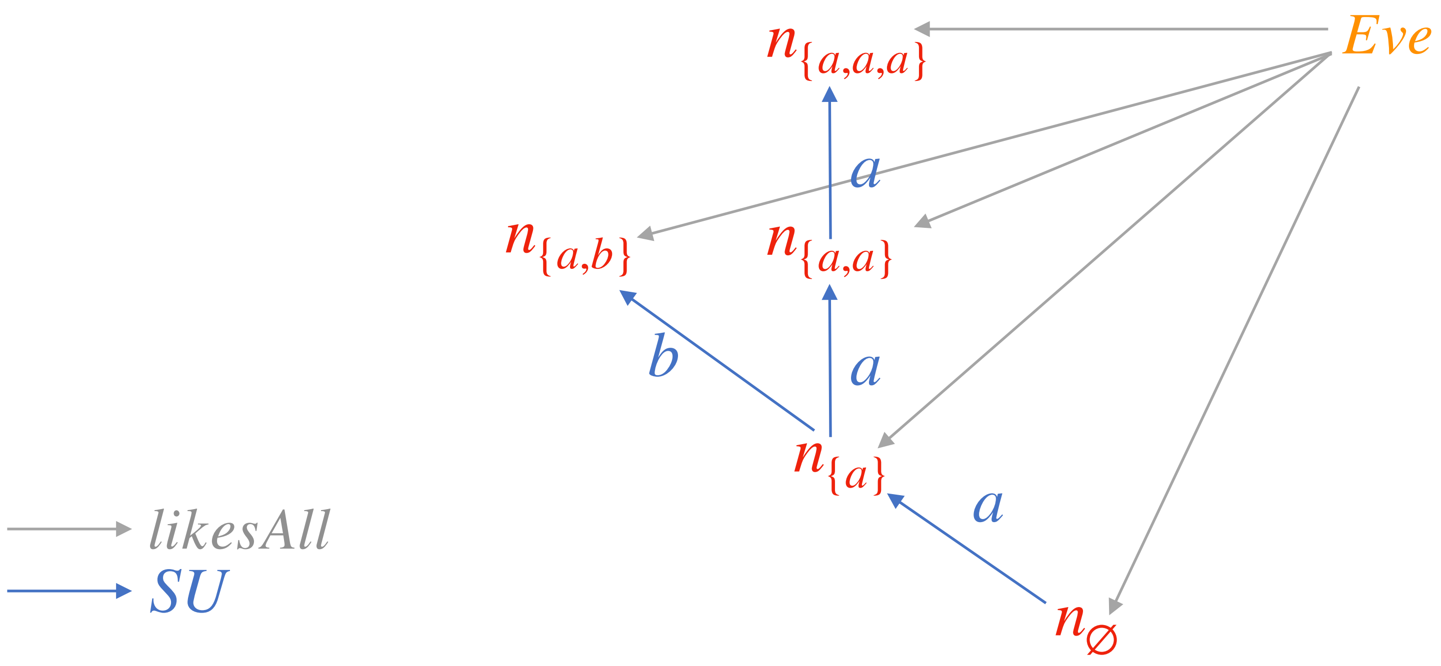
$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V . empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V . likesAll(x, V) \wedge SU(S, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

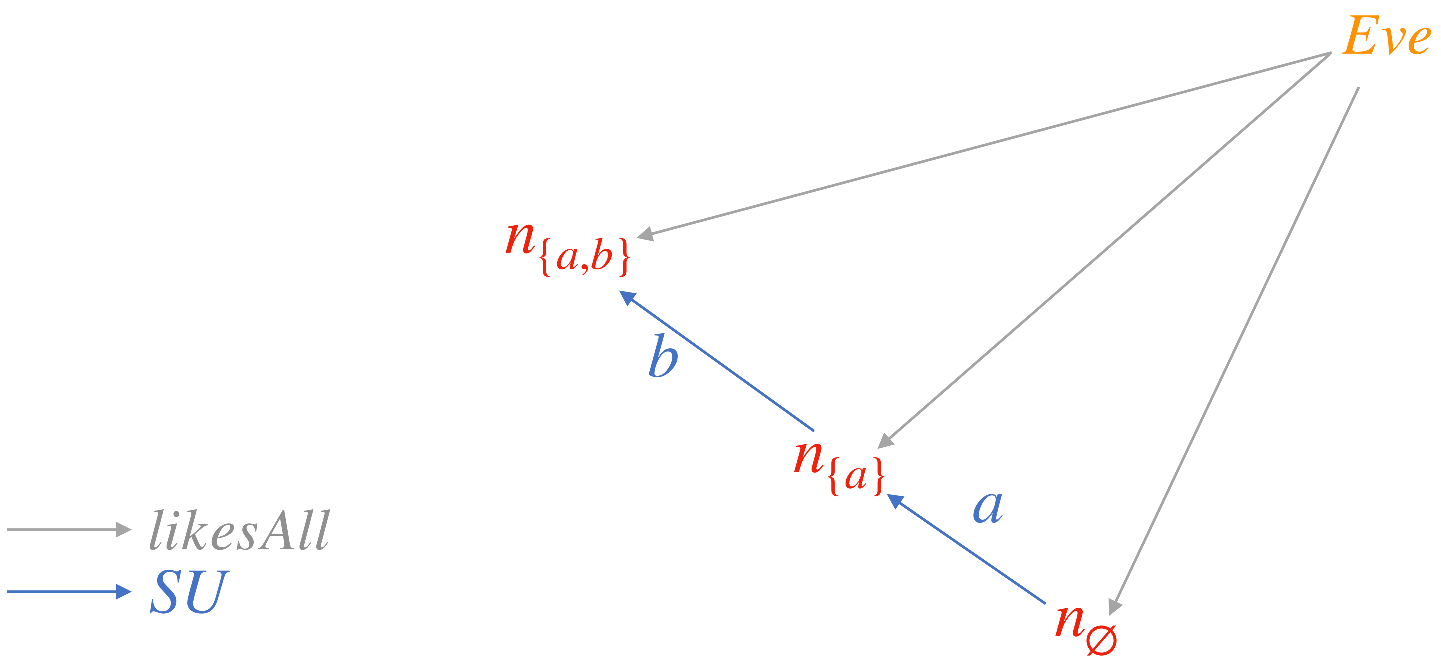
$$person(x) \to likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \land likes(x, y) \to likesAll(x, S \cup \{y\}) \quad (2)$$

$$\to \exists V . empty(V) \quad (1')$$

$$person(x) \land empty(Y) \to likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \land likes(x, y) \to \exists V . likesAll(x, V) \land SU(S, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

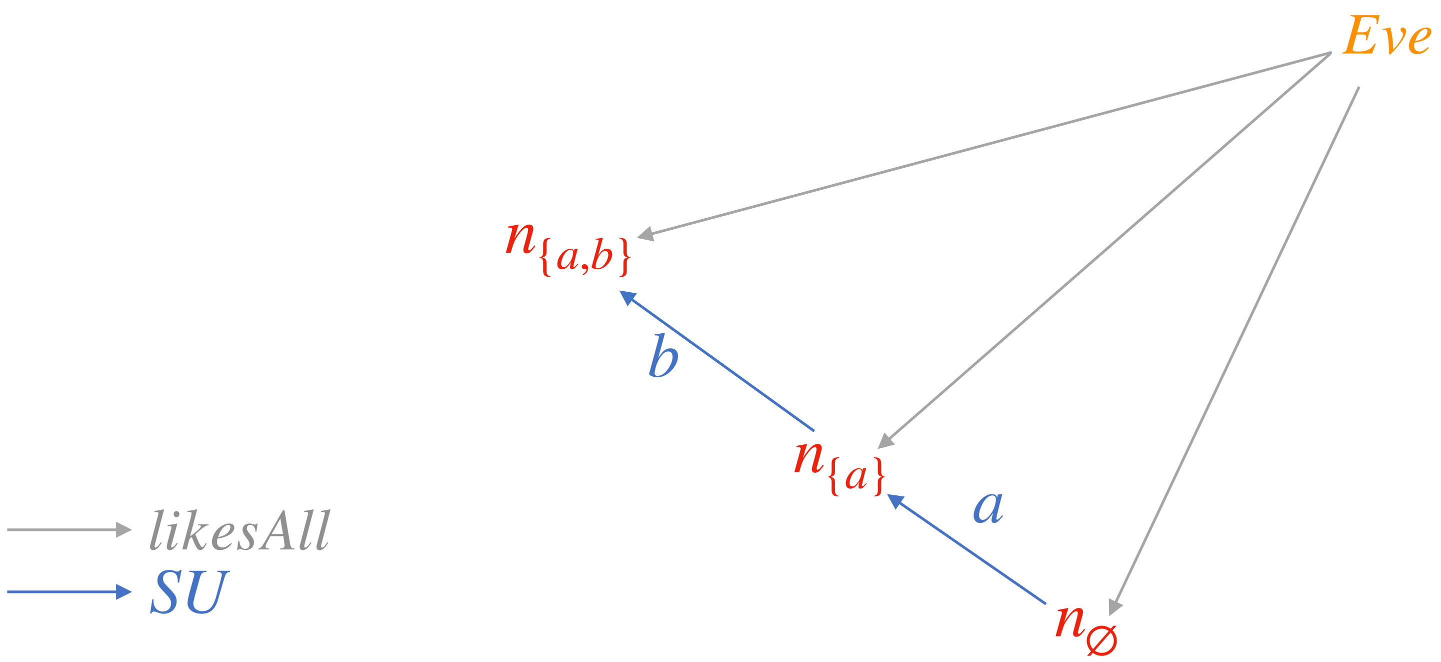
$$person(x) \to likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \land likes(x, y) \to likesAll(x, S \cup \{y\}) \quad (2)$$

$$\to \exists V . empty(V) \quad (1')$$

$$person(x) \land empty(Y) \to likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \land likes(x, y) \to \exists V . likesAll(x, V) \land SU(S, y, V) \land SU(V, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

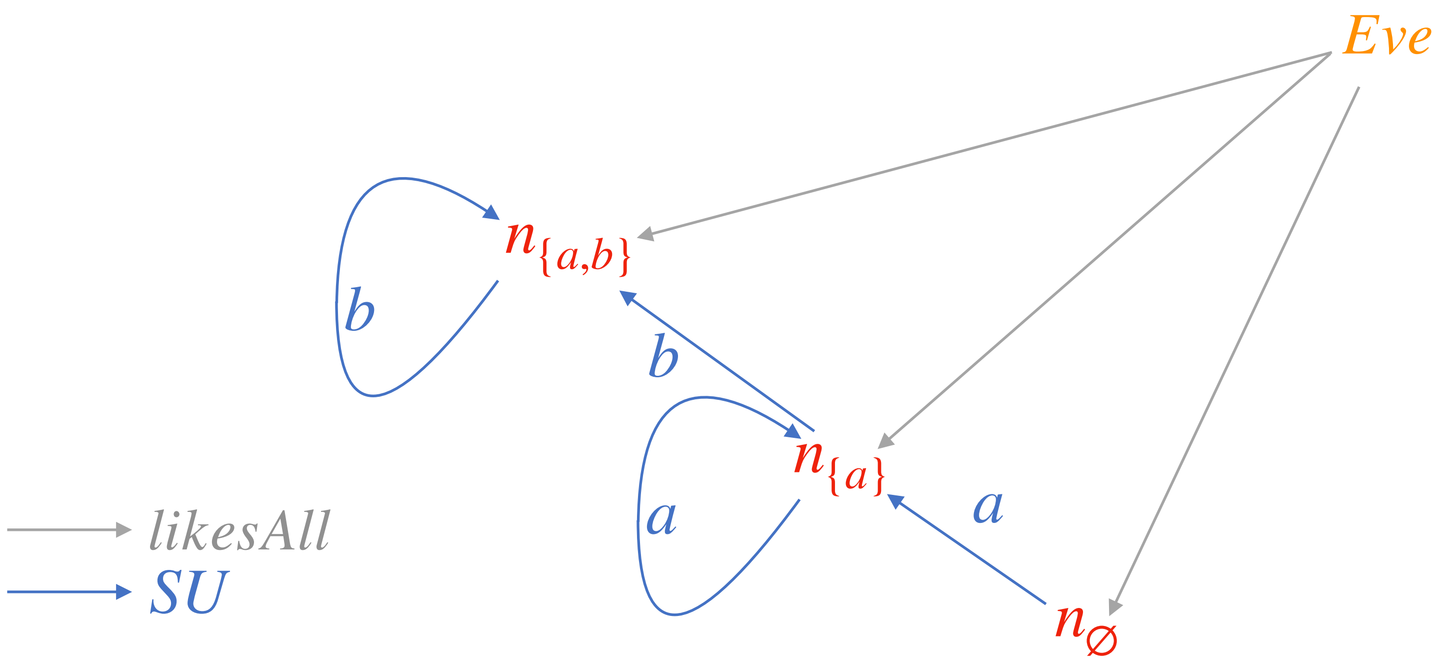
$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V . empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V . likesAll(x, V) \wedge SU(S, y, V) \wedge SU(V, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

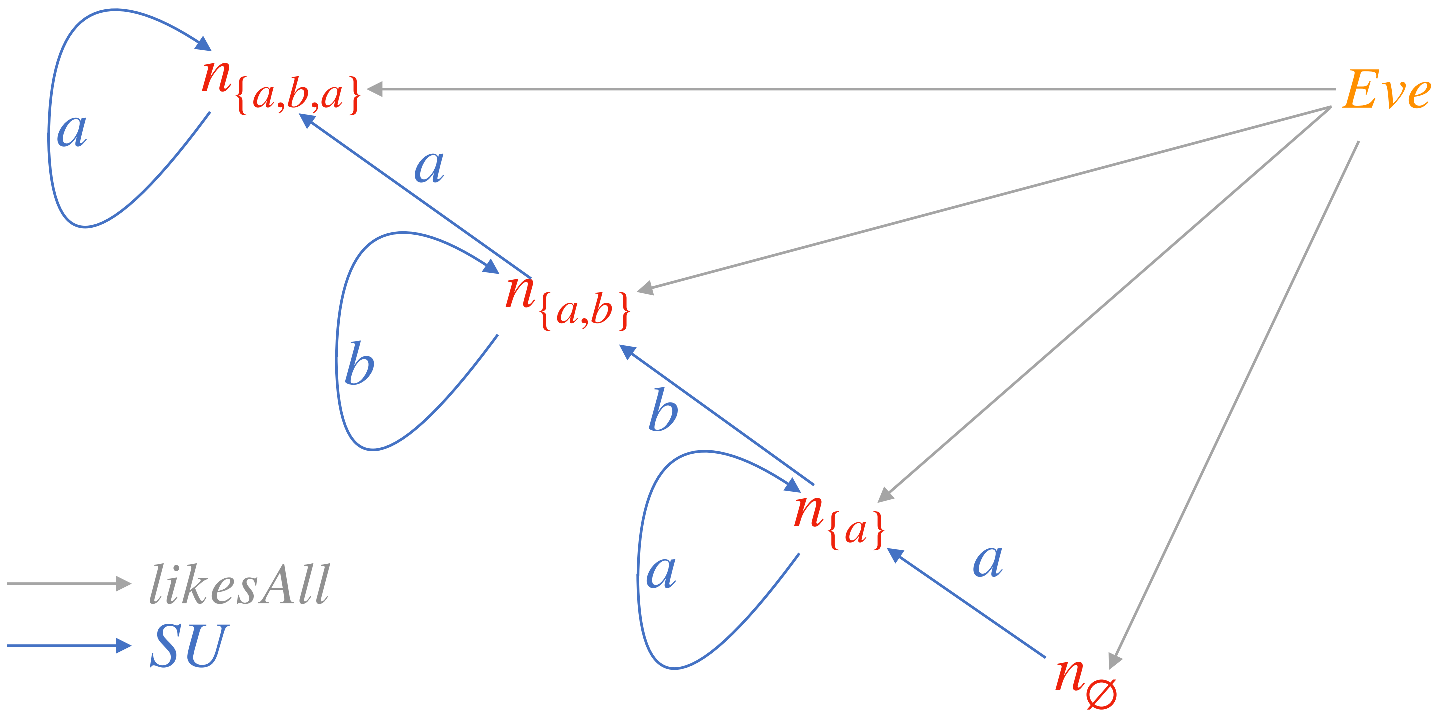
$$person(x) \to likesAll(x, \emptyset) \quad (1)$$

$$likesAll(x, S) \land likes(x, y) \to likesAll(x, S \cup \{y\}) \quad (2)$$

$$\to \exists V \,.\, empty(V) \quad (1')$$

$$person(x) \land empty(Y) \to likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \land likes(x, y) \to \exists V \,.\, likesAll(x, V) \land SU(S, y, V) \land SU(V, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

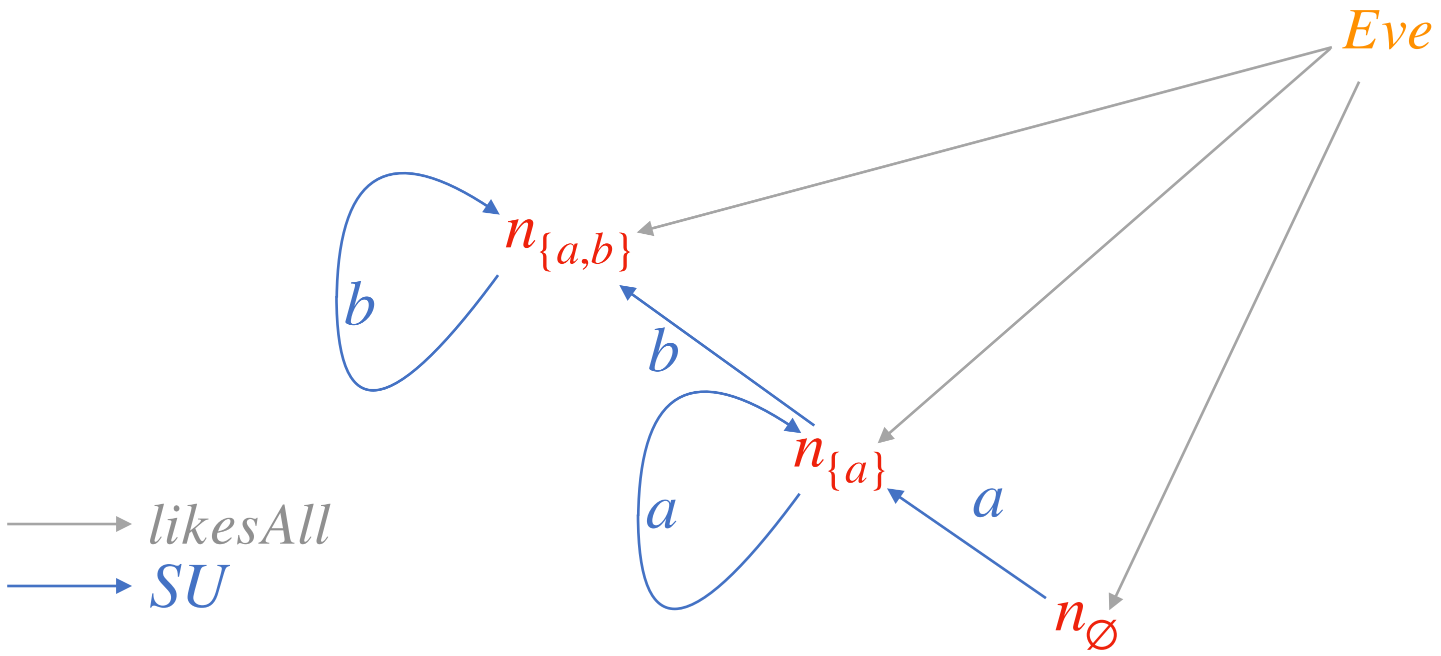
$$person(x) \to likesAll(x, \varnothing) \quad (1)$$

$$likesAll(x, S) \land likes(x, y) \to likesAll(x, S \cup \{y\}) \quad (2)$$

$$\to \exists V . empty(V) \quad (1')$$

$$person(x) \land empty(Y) \to likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \land likes(x, y) \to \exists V . likesAll(x, V) \land SU(S, y, V) \land SU(V, y, V) \quad (2')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

$$person(x) \rightarrow likesAll(x, \emptyset) \quad (1)$$

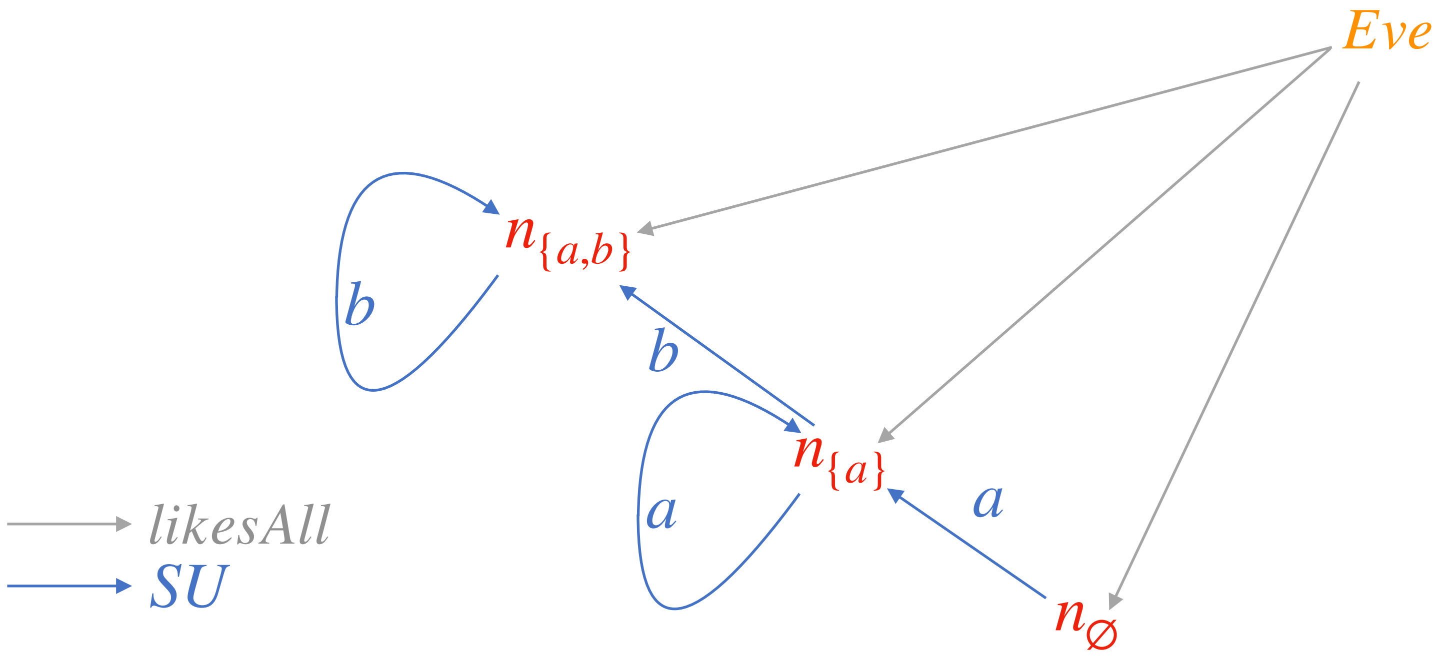
$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \quad (2)$$

$$\rightarrow \exists V . empty(V) \quad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \quad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V . likesAll(x, V) \wedge SU(S, y, V) \wedge SU(V, y, V) \quad (2')$$

$$SU(U, x, V) \wedge SU(U, y, U) \rightarrow SU(V, y, V) \quad (2'')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

$$person(x) \rightarrow likesAll(x, \emptyset) \qquad (1)$$

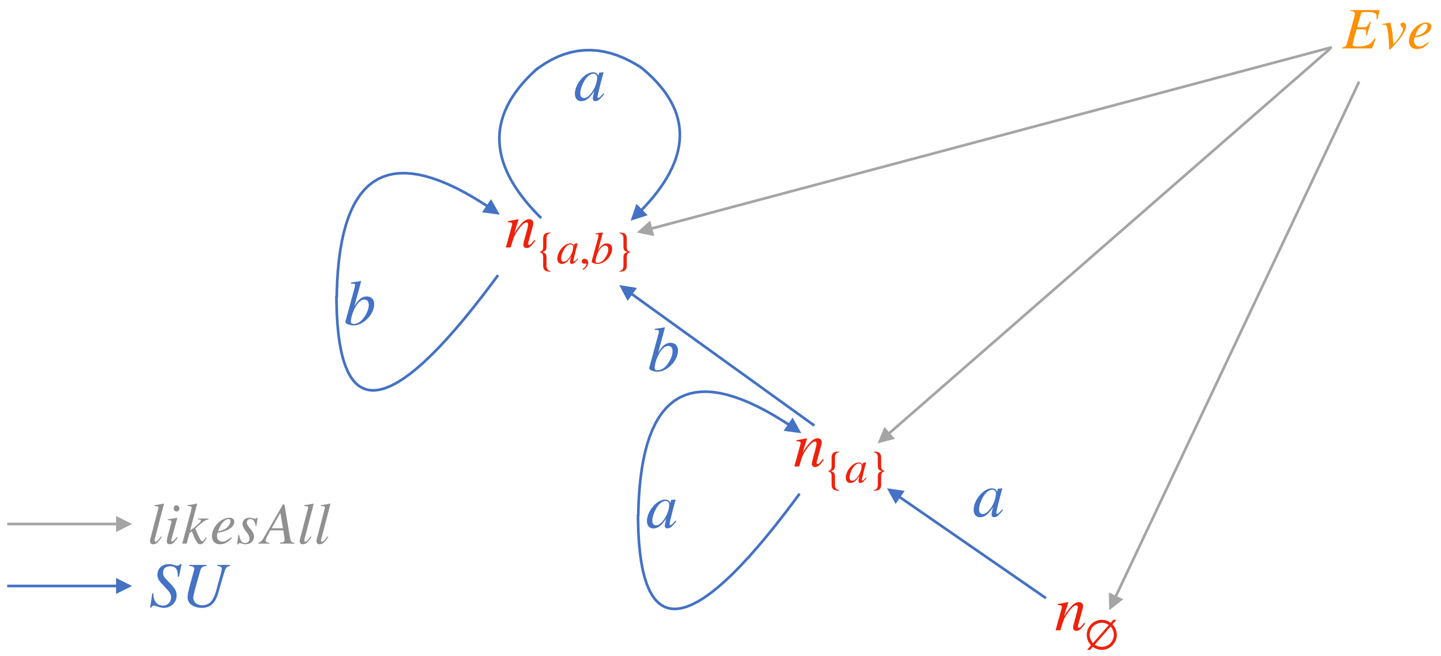
$$likesAll(x, S) \wedge likes(x, y) \rightarrow likesAll(x, S \cup \{y\}) \qquad (2)$$

$$\rightarrow \exists V . empty(V) \qquad (1')$$

$$person(x) \wedge empty(Y) \rightarrow likesAll(x, Y) \qquad (1'')$$

$$likesAll(x, S) \wedge likes(x, y) \rightarrow \exists V . likesAll(x, V) \wedge SU(S, y, V) \wedge SU(V, y, V) \qquad (2')$$

$$SU(U, x, V) \wedge SU(U, y, U) \rightarrow SU(V, y, V) \qquad (2'')$$



$person(Eve)$

$likes(Eve, a)$

$likes(Eve, b)$

Datalog(S) to existential rules

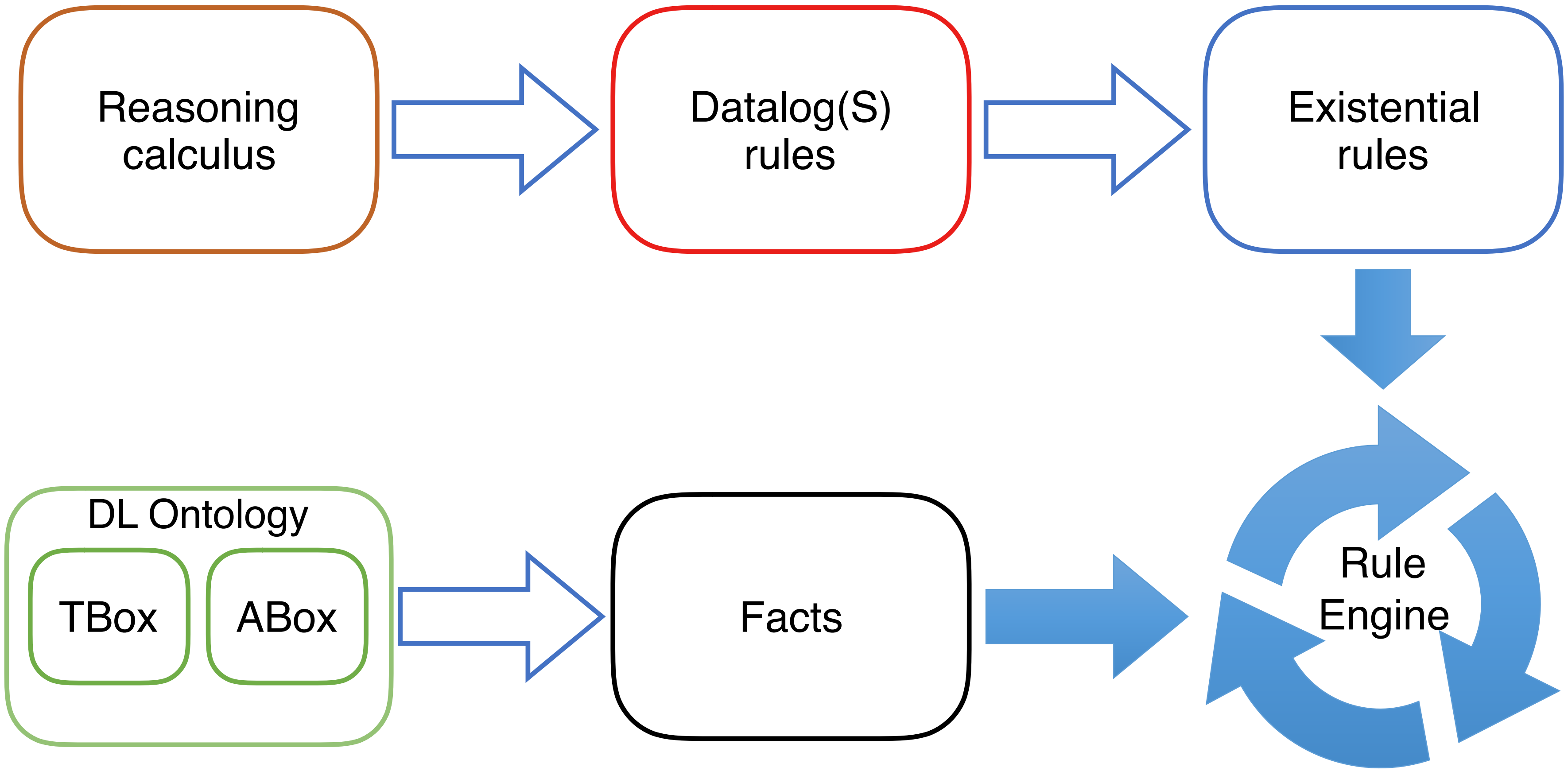
Theorem: Any Datalog(S) rule set can be

- 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 $\mathcal{O}$, we obtain an ontology $\mathcal{O}'$ containing only concept inclusions of the form $A_1 \sqsubseteq A_2$, $A \sqsubseteq \mathsf{st}(C_+)$, and $\mathsf{st}(C_-) \sqsubseteq A$, where C_+ occurs positively in $\mathcal{O}$ and C_- occurs negatively in $\mathcal{O}$. Since $\mathcal{O}$ is a Horn $\mathcal{SHIQ}$ ontology, C_+ can only be of the form $\top$, $\bot$, A, $\neg C$, $C \sqcap D$, $\exists R.C$, $\forall R.C$, $\geqslant nS.C$, or $\leqslant 1S.C$, and C_- only of the form $\top$, $\bot$, A, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, or $\geqslant 1R.C$.

Concept inclusions of the form $A \sqsubseteq \mathsf{st}(C_+)$ that are not of form (n1), are transformed to form (n1) as follows:

- $A \sqsubseteq \mathsf{st}(\neg C) = \neg A_C \;\Rightarrow\; A \sqcap A_C \sqsubseteq \bot$;
- $A \sqsubseteq \mathsf{st}(\geqslant nS.C) = \geqslant nS.A_C \;\Rightarrow\; A \sqsubseteq \exists S.B_i$, $B_i \sqsubseteq A_C$, $1 \le i \le n$, $B_i \sqcap B_j \sqsubseteq \bot$, $1 \le i < j \le n$, where B_i are fresh atomic concepts.

Concept inclusions of the form $\mathsf{st}(C_-) \sqsubseteq A$ that are not of form (n1) are transformed to form (n1) as follows:

- $\mathsf{st}(C \sqcup D) = A_C \sqcup A_D \sqsubseteq A \;\Rightarrow\; A_C \sqsubseteq A$, $A_D \sqsubseteq A$;
- $\mathsf{st}(\exists R.C) = \exists R.A_C \sqsubseteq A \;\Rightarrow\; A_C \sqsubseteq \forall R^-.A$;
- $\mathsf{st}(\geqslant 1S.C) = \geqslant 1S.A_C \sqsubseteq A \;\Rightarrow\; A_C \sqsubseteq \forall S^-.A$.

It is easy to show using Proposition 1, that $\mathcal{O}' \models \alpha$ iff $\mathcal{O} \models \alpha$ for every axiom α 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

I1 $$\frac{}{M \sqcap A \sqsubseteq A}$$ **I2** $$\frac{}{M \sqsubseteq \top}$$

R1 $$\frac{M \sqsubseteq A_i \quad \bigsqcap A_i \sqsubseteq C \in \mathcal{O}}{M \sqsubseteq C}$$

R2 $$\frac{M \sqsubseteq \exists R.N \quad N \sqsubseteq \bot}{M \sqsubseteq \bot}$$

R3 $$\frac{M \sqsubseteq \exists R_1.N \quad M \sqsubseteq \forall R_2.A \quad R_1 \sqsubseteq_{\mathcal{O}} R_2}{M \sqsubseteq \exists R_1.(N \sqcap A)}$$

R4 $$\frac{M \sqsubseteq \exists R_1.N \quad N \sqsubseteq \forall R_2.A \quad R_1 \sqsubseteq_{\mathcal{O}} R_2^-}{M \sqsubseteq A}$$

R5 $$\frac{\begin{array}{lll} M \sqsubseteq \exists R_1.N_1 & N_1 \sqsubseteq B & R_1 \sqsubseteq_{\mathcal{O}} S \\ M \sqsubseteq \exists R_2.N_2 & N_2 \sqsubseteq B & R_2 \sqsubseteq_{\mathcal{O}} S \\ M \sqsubseteq \leqslant 1S.B & & \end{array}}{M \sqsubseteq \exists R_1.(N_1 \sqcap N_2)}$$

R6 $$\frac{\begin{array}{lll} M \sqsubseteq \exists R_1.N_1 & N_1 \sqsubseteq \exists R_2.(N_2 \sqcap A) & R_1 \sqsubseteq_{\mathcal{O}} S^- \\ M \sqsubseteq B & N_2 \sqcap A \sqsubseteq B & R_2 \sqsubseteq_{\mathcal{O}} S \\ N_1 \sqsubseteq \leqslant 1S.B & & \end{array}}{M \sqsubseteq A \quad M \sqsubseteq \exists R_2^-.N_1}$$

Table 3: Saturation Rules for Horn $\mathcal{SHIQ}$ Ontologies

Consequence-driven classification

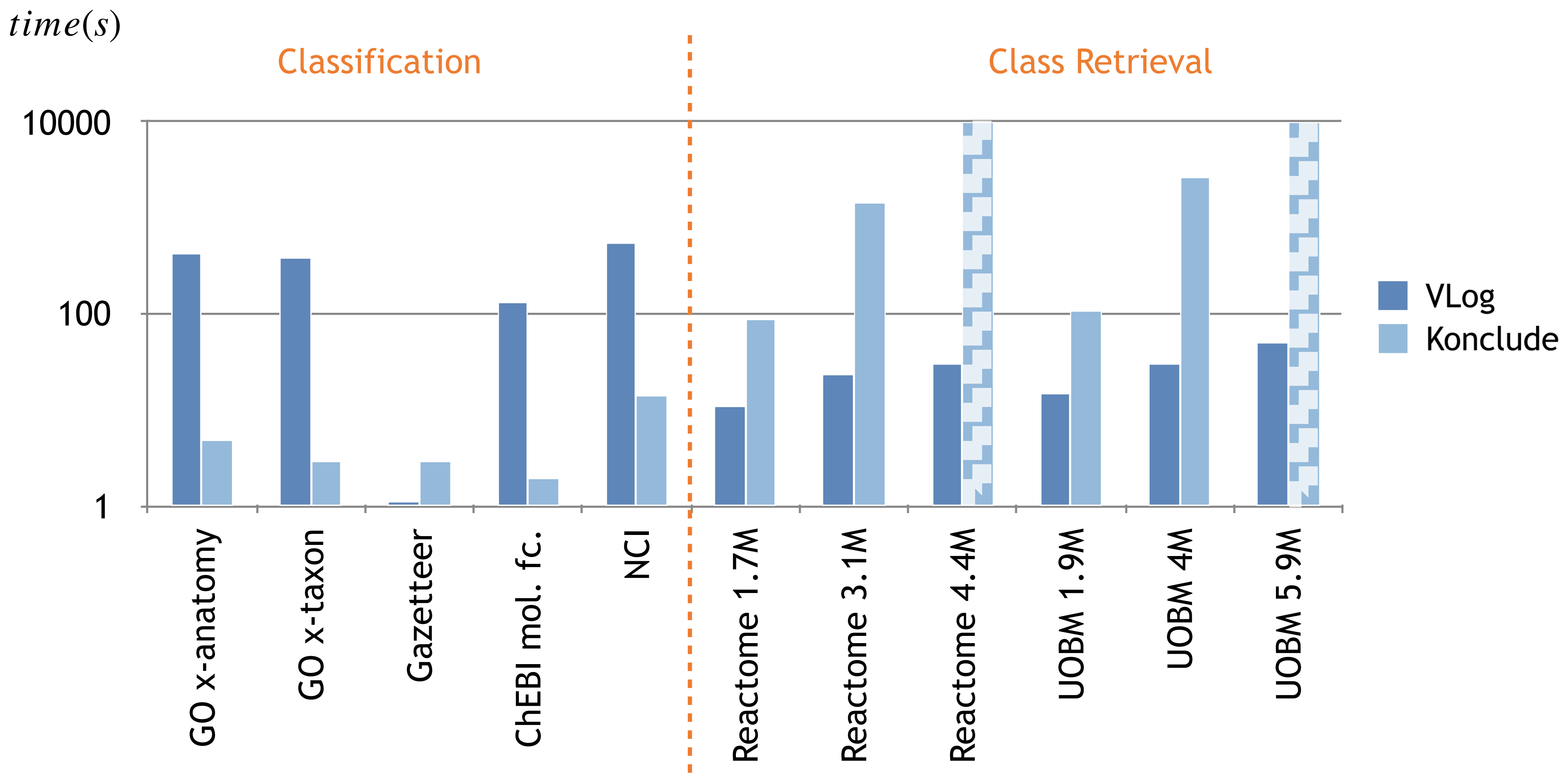
$$\frac{H \sqsubseteq \exists R\,.\,K \quad H \sqsubseteq A}{H \sqsubseteq \exists R\,.\,(K \sqcap B)} : A \sqsubseteq \forall R\,.\,B \in \mathcal{O}$$

Consequence-driven classification

$$\frac{H \sqsubseteq \exists R \,.\, K \quad H \sqsubseteq A}{H \sqsubseteq \exists R \,.\, (K \sqcap B)} : A \sqsubseteq \forall R \,.\, B \in \mathcal{O}$$

$$\mathsf{Exists}(H, r, K) \land \mathsf{SubClass}(H, a) \land \mathsf{ax}_{\sqsubseteq\forall}(a, r, b)$$
$$\rightarrow \mathsf{Exists}(H, r, K \cup \{b\})$$

Evaluation


time(s)
Classification
Class Retrieval
10000
100
1
GO x-anatomy
GO x-taxon
Gazetteer
ChEBI mol. fc.
NCI
Reactome 1.7M
Reactome 3.1M
Reactome 4.4M
UOBM 1.9M
UOBM 4M
UOBM 5.9M
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