Chasing Sets: How to Use Existential Rules for Expressive Reasoning

David Carral, Irina Dragostea, 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

\[
person(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1)
\]
\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2)
\]
Surface language for existential rules with terminating chase

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

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

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

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

\[
\begin{align*}
\text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
\text{likesAll}(x, S) & \rightarrow \text{allLikeAll}({x}, S) \quad (3) \\
\text{allLikeAll}(S, T) \land \text{likesAll}(x, T) & \rightarrow \text{allLikeAll}(S \cup \{x\}, T) \quad (4) \\
\text{allLikeAll}(S, S) \land \text{alice} \in S & \rightarrow \text{cliqueOfAlice}(S) \quad (5)
\end{align*}
\]
Datalog(S): Definition

Logic with two sorts: \textbf{objects} and \textbf{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.
\[
\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \\ (2)
\]
\[ \text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \]
\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \]
\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]
\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]
person(x) → likesAll(x, ∅) \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(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)
\[
\text{person}(x) \rightarrow \text{likesAll}(x, \emptyset) \quad (1)
\]
\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2)
\]
\[
\rightarrow \exists V. \text{empty}(V) \quad (1')
\]
\[
\text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'')
\]
\[
\text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \quad (2')
\]

\[
Eve
\]

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

\textit{Eve}

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

\[
\rightarrow \exists V. \text{empty}(V) \quad (1')
\]

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

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

\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]

\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]

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

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

\[ \rightarrow \exists V. \text{empty}(V) \quad (1') \]

\[ \text{person}(x) \land \text{empty}(Y) \rightarrow \text{likesAll}(x, Y) \quad (1'') \]

\[ \text{likesAll}(x, S) \land \text{likes}(x, y) \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2') \]
\[
\begin{align*}
   \text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
   \text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
   & \rightarrow \exists V. \text{empty}(V) \quad (1') \\
   \text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
   \text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2')
\end{align*}
\]
\[
\begin{align*}
\text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
& \rightarrow \exists V. \text{empty}(V) \quad (1') \\
\text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \exists V. \text{likesAll}(x, V) \land SU(S, y, V) \land SU(V, y, V) \quad (2') 
\end{align*}
\]
\[\begin{align*}
\text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
& \rightarrow \exists V. \text{empty}(V) \quad (1') \\
\text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2') \\
\text{SU}(U, x, V) \land \text{SU}(U, y, U) & \rightarrow \text{SU}(V, y, V) \quad (2'')
\end{align*}\]
\[
\begin{align*}
\text{person}(x) & \rightarrow \text{likesAll}(x, \emptyset) \quad (1) \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \text{likesAll}(x, S \cup \{y\}) \quad (2) \\
& \rightarrow \exists V. \text{empty}(V) \quad (1') \\
\text{person}(x) \land \text{empty}(Y) & \rightarrow \text{likesAll}(x, Y) \quad (1'') \\
\text{likesAll}(x, S) \land \text{likes}(x, y) & \rightarrow \exists V. \text{likesAll}(x, V) \land \text{SU}(S, y, V) \land \text{SU}(V, y, V) \quad (2') \\
\text{SU}(U, x, V) \land \text{SU}(U, y, U) & \rightarrow \text{SU}(V, y, V) \quad (2'')
\end{align*}
\]
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 $A_1 \sqsubseteq A_2$, $A \sqsubseteq st(C)$, and $st(C) \sqsubseteq A$, where $C$ occurs positively in $O$ and $C'$ occurs negatively in $O$. Since $O$ is a Horn SHIQ ontology, $C$ can only be of the form $T$, $\bot$, $A$, $\neg C$, $C \sqsubseteq D$, $\exists R.C$, $\forall R.C$, $\ni n.S.C$, or $\ni l.S.C$, and $C'$ only of the form $T$, $\bot$, $A$, $C \sqsubseteq D$, $C \sqsubseteq D$, $\exists R.C$, or $\forall R.C$.

Concept inclusions of the form $A \sqsubseteq st(C)$ that are not of form (n1), are transformed to form (n1) as follows:

- $A \sqsubseteq st(C) = C \sqsubseteq A \sqsubseteq C$.
- $A \sqsubseteq st(\ni n.S.C) = \ni n.S.A \sqsubseteq C \sqsubseteq A \sqsubseteq \exists S.B_i$, $\exists S.B_i \subseteq A$, $1 \leq i \leq n$, $B_i \sqsubseteq B_j \sqsubseteq \bot$, $1 \leq i < j \leq n$, where $B_i$ are fresh atomic concepts.

Concept inclusions of the form $st(C) \subseteq A$ that are not of form (n1) are transformed to form (n1) as follows:

- $st(C \sqcup D) = A \sqcup A \sqsubseteq A \sqsubseteq A \sqcup A \sqsubseteq A$.
- $st(\forall R.C) = \forall R.A \sqsubseteq A \sqsubseteq A \sqsubseteq \forall R^- A$.
- $st(\ni l.S.C) = \ni l.S.A \sqsubseteq A \sqsubseteq A \sqsubseteq \forall S^- A$.

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

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 7 can interact only with axioms.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Premise</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>$M \sqsubseteq A_i \sqsubseteq A \sqsubseteq C$</td>
<td>$M \sqsubseteq C$</td>
</tr>
<tr>
<td>R2</td>
<td>$M \sqsubseteq \exists R.N \sqsubseteq N \sqsubseteq \bot$</td>
<td>$M \sqsubseteq \bot$</td>
</tr>
<tr>
<td>R3</td>
<td>$M \sqsubseteq \exists R_1.N \sqsubseteq R_2.A \sqsubseteq R_1 \sqsubseteq \forall R_2.A$</td>
<td>$M \sqsubseteq \exists R_1.(N \sqsubseteq A)$</td>
</tr>
<tr>
<td>R4</td>
<td>$M \sqsubseteq \exists R_1.N \sqsubseteq \forall R_2.A \sqsubseteq R_1 \sqsubseteq \forall R_2.\neg$</td>
<td>$M \sqsubseteq \exists R_1.(N \sqsubseteq A)$</td>
</tr>
<tr>
<td>R5</td>
<td>$M \sqsubseteq \exists R_1.N \sqsubseteq B \sqsubseteq \forall R_2.A \sqsubseteq R_1 \sqsubseteq \forall R_2.\neg$</td>
<td>$M \sqsubseteq \exists R_1.(N \sqsubseteq A)$</td>
</tr>
<tr>
<td>R6</td>
<td>$M \sqsubseteq \exists R_1.N \sqsubseteq B \sqsubseteq \forall R_2.A \sqsubseteq R_1 \sqsubseteq \forall R_2.\neg$</td>
<td>$M \sqsubseteq \exists R_1.(N \sqsubseteq A)$</td>
</tr>
</tbody>
</table>

Table 3: Saturation Rules for Horn SHIQ Ontologies
Consequence-driven classification

\[
\begin{array}{c}
H \subseteq \exists R . K \\
H \subseteq A
\end{array}
\quad \quad
\Rightarrow
\quad
\begin{array}{c}
H \subseteq \exists R . (K \cap B) \\
A \subseteq \forall R . B \in \emptyset
\end{array}
\]
Consequence-driven classification

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

Exists($H$, $r$, $K$) $\land$ SubClass($H$, $a$) $\land$ ax$_{\subseteq \forall}$(a, $r$, $b$)

$\rightarrow$ Exists($H$, $r$, $K \cup \{b\}$)
Evaluation

![Evaluation Chart]

- **Classification**
  - GO x-anatomy
  - GO x-taxon
  - Gazetteer
  - ChEBI mol. f.c.
  - NCI
  - Reactome 1.7M
  - Reactome 3.1M
  - Reactome 4.4M

- **Class Retrieval**
  - UOBM 1.9M
  - UOBM 4M
  - UOBM 5.9M

*time(s)*

- 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