Expressivity of Planning with Horn Description Logic Ontologies

AAAI'22, virtual, 22th February – 1st March, 2022
Planning and Reasoning

AI planning

- actions that modify abstract states
- fixed domain
- closed-world assumption
Planning and Reasoning

AI planning
- actions that modify abstract states
- fixed domain
- closed-world assumption

+ Static reasoning
- global state constraints
- high-level background knowledge
- open-world assumption
Planning and Reasoning

AI planning
- actions that modify abstract states
- fixed domain
- closed-world assumption

+ Static reasoning
- global state constraints
- high-level background knowledge
- open-world assumption

explicit-input knowledge and action bases (eKABs)
(Calvanese, Montali, Patrizi, and Stawowy 2016)
Planning and Reasoning

AI planning
- actions that modify abstract states
- fixed domain
- closed-world assumption

+ Static reasoning
- global state constraints
- high-level background knowledge
- open-world assumption

explicit-input knowledge and action bases (eKABs)
(Calvanese, Montali, Patrizi, and Stawowy 2016)

Making eKABs more expressive and practical
Outline

Preliminaries

- Description logics (DLs)
- Planning domain definition language (PDDL)
- Explicit-input knowledge and action bases (eKABs)

Results

- Compilability
- Non-compilability
- Experiments
Description logics (DLs) – A simple example

**Description logics**: decidable fragments of FOL

TBox $\mathcal{T} = \{ \text{Drone} \sqsubseteq \text{MovingObject},$

\hspace{1cm} \text{Drone} \sqcap \exists \text{near}.\text{MovingObject} \sqsubseteq \text{RiskOfPhysicalDamage},$

\hspace{1cm} \text{Drone} \sqsubseteq \exists \text{controlledBy}.\text{Operator} \sqcup \text{AI} \}$

state $s = \{ \text{Drone}(a), \text{Drone}(b), \text{near}(a, b) \}$
Description logics (DLs) – A simple example

**Description logics**: decidable fragments of FOL

\[ \text{TBox } T = \{ \text{Drone } \sqsubseteq \text{MovingObject}, \]
\[ \text{Drone } \sqcap \exists \text{near.} \text{MovingObject } \sqsubseteq \text{RiskOfPhysicalDamage}, \]
\[ \text{Drone } \sqsubseteq \exists \text{controlledBy.} \text{Operator } \sqcup \text{AI } \} \]

**state** \( s = \{ \text{Drone}(a), \text{Drone}(b), \text{near}(a, b) \} \)

Some DLs: DL-Lite, \( \mathcal{ELH} \perp \), Horn-\( \text{ALCHOIQ} \), Horn-\( \text{SHIQ} \), Horn-\( \text{SROIQ} \), \( \text{ALCI} \), \( \text{SH} \), …
Description logics (DLs) – A simple example

**Description logics:** decidable fragments of FOL

TBox $\mathcal{T} = \{ \text{Drone} \sqsubseteq \text{MovingObject},$
$\text{Drone} \sqcap \exists \text{near}.\text{MovingObject} \sqsubseteq \text{RiskOfPhysicalDamage},$
$\text{Drone} \sqsubseteq \exists \text{controlledBy}.\text{Operator} \sqcup \text{AI} \}$

state $s = \{ \text{Drone}(a), \text{Drone}(b), \text{near}(a, b) \}$

Some DLs: DL-Lite, $\mathcal{ELH}_{\bot}$, Horn-$\mathcal{ALCHOIQ}$, Horn-$\mathcal{SHIQ}$, Horn-$\mathcal{SROIQ}$, $\mathcal{ALCI}$, $\mathcal{SH}$, ...

Conjunctive queries (CQs):

$s, \mathcal{T} \models \exists x, y. \text{RiskOfPhysicalDamage}(x) \land \text{RiskOfPhysicalDamage}(y) \land \text{near}(x, y)$
Datalog rewritings for CQs over DLs

Datalog\(^{\neg}\) rewriting \(R_{\mathcal{T},Q}\):

\[
\begin{align*}
\text{MovingObject}(x) & \leftarrow \text{Drone}(x) \\
\text{RiskOfPhysicalDamage}(x) & \leftarrow \text{Drone}(x) \land \text{near}(x, y) \land \text{MovingObject}(y) \\
Q & \leftarrow \text{RiskOfPhysicalDamage}(x) \land \text{RiskOfPhysicalDamage}(y) \land \text{near}(x, y)
\end{align*}
\]
Datalog rewritings for CQs over DLs

Datalog(¬) rewriting \( R_{T,Q} \):

MovingObject\( (x) \leftarrow \) Drone\( (x) \)
RiskOfPhysicalDamage\( (x) \leftarrow \) Drone\( (x) \) \( \land \) near\( (x, y) \) \( \land \) MovingObject\( (y) \)
\( Q \leftarrow \) RiskOfPhysicalDamage\( (x) \) \( \land \) RiskOfPhysicalDamage\( (y) \) \( \land \) near\( (x, y) \)

- exist for \( \mathcal{ELH}_\perp \) (polynomial), Horn-\( \mathcal{SHIQ} \) (exponential), …
- reduce open-world to closed-world reasoning
Planning domain definition language (PDDL)

(:init (Drone a) (Drone b) (near a b) . . . )

• closed-world states
• /uniFB01rst-order formulas in conditions and action effects
• model checking is decidable
• task: find a plan to reach the goal from the initial state
Planning domain definition language (PDDL)

(:init (Drone a) (Drone b) (near a b) ...)

(:action Move
    :parameters (?x ?y)
    :precondition (and (Drone ?x) (near ?x ?y) (not (Object ?y)))
    :effect (and (not (Drone ?x)) (Drone ?y))
  )

• closed-world states
• first-order formulas in conditions and action effects
• model checking is decidable
• task: find a plan to reach the goal from the initial state
Planning domain definition language (PDDL)

(:init (Drone a) (Drone b) (near a b) ...)

(:action Move
   :parameters (?x ?y)
   :precondition (and (Drone ?x) (near ?x ?y) (not (Object ?y)))
   :effect (and (not (Drone ?x)) (Drone ?y))
)

(:goal
   (not (exists (?x ?y) (and (Drone ?x) (Drone ?y) (near ?x ?y))))
)
Planning domain definition language (PDDL)

(:init (Drone a) (Drone b) (near a b) ...)

(:action Move
  :parameters (?x ?y)
  :precondition (and (Drone ?x) (near ?x ?y) (not (Object ?y)))
  :effect (and (not (Drone ?x)) (Drone ?y))
)

(:goal
  (not (exists (?x ?y) (and (Drone ?x) (Drone ?y) (near ?x ?y)))))
)

- closed-world states
- first-order formulas in conditions and action effects
- model checking is decidable
- task: find a plan to reach the goal from the initial state
eKABs

Explicit-input knowledge and action bases (Calvanese, Montali, Patrizi, and Stawowy 2016)

“PDDL + TBox + epistemic conjunctive queries (ECQs)”
eKABs

Explicit-input knowledge and action bases (Calvanese, Montali, Patrizi, and Stawowy 2016)

“PDDL + TBox + epistemic conjunctive queries (ECQs)”

Drone $\sqcap \exists$ near.MovingObject $\sqsubseteq$ RiskOfPhysicalDamage

(:goal (not (mko (exists (?x ?y) (and
    (RiskOfPhysicalDamage ?x)
    (RiskOfPhysicalDamage ?y)
    (near ?x ?y)
  ))))))

- epistemic minimal-knowledge operator (mko)
- open-world states and conditions, but closed-world effects
eKABs

Explicit-input knowledge and action bases (Calvanese, Montali, Patrizi, and Stawowy 2016)

“PDDL + TBox + epistemic conjunctive queries (ECQs)”

Drone ⊓∃near.MovingObject ⊑ RiskOfPhysicalDamage

(:goal (not (mko (exists (?x ?y) (and (RiskOfPhysicalDamage ?x) (RiskOfPhysicalDamage ?y) (near ?x ?y)))))

- epistemic minimal-knowledge operator (mko)
- open-world states and conditions, but closed-world effects
- split modeling into static and dynamic part
- complexity of checking entailment of ECQs depends on the TBox
Outline

Preliminaries
- Description logics (DLs)
- Planning domain definition language (PDDL)
- Explicit-input knowledge and action bases (eKABs)

Results
- Compilability
- Non-compilability
- Experiments
Compiling eKABs into PDDL

(Calvanese, Montali, Patrizi, and Stawowy 2016):

- compilation of DL-Lite eKABs into PDDL
- **key**: rewriting (E)CQs into first-order formulas
- however, resulting PDDL exponentially larger
- can be made practical (Borgwardt, Hoffmann, Kovtunova, and Steinmetz 2021)
Compiling eKABs into PDDL

(Calvanese, Montali, Patrizi, and Stawowy 2016):

• compilation of DL-Lite eKABs into PDDL
• key: rewriting (E)CQs into first-order formulas
• however, resulting PDDL exponentially larger
• can be made practical (Borgwardt, Hoffmann, Kevtunova, and Steinmetz 2021)

Here:

• more expressive (Horn-)DLs
• polynomial-size compilation schemes that preserve plan size polynomially
  (Thiébaux, Hoffmann, and Nebel 2005)
• compilation may take super-polynomial time
• use (polynomial-size) Datalog \neg rewritings of (E)CQs
Main Compilability Result

Theorem

For the class of eKAB instances for which all ECQs are Datalog rewritable w.r.t. the TBox, there is a compilation scheme into PDDL that preserves plan size exactly.
Main Compilability Result

Theorem

For the class of eKAB instances for which all ECQs are Datalog\(^\neg\) rewritable w.r.t. the TBox, there is a compilation scheme into PDDL that preserves plan size exactly.

- use derived predicates to include Datalog\(^\neg\) rules in PDDL 2.2:

  \[
  \text{RiskOfPhysicalDamage}(x) \leftarrow \text{Drone}(x) \land \text{near}(x, y) \land \text{MovingObject}(y)
  \]

  (:derived (RiskOfPhysicalDamage ?x)
   (exists ?y (and (Drone ?x) (near ?x ?y) (MovingObject ?y)))
  )
Main Compilability Result

Theorem

For the class of eKAB instances for which all ECQs are Datalog$\neg$ rewritable w.r.t. the TBox, there is a compilation scheme into PDDL that preserves plan size exactly.

- use derived predicates to include Datalog$\neg$ rules in PDDL 2.2:
  
  \[
  \text{RiskOfPhysicalDamage}(x) \leftarrow \text{Drone}(x) \land \text{near}(x, y) \land \text{MovingObject}(y)
  \]
  
  (:derived (RiskOfPhysicalDamage ?x) (exists ?y (and (Drone ?x) (near ?x ?y) (MovingObject ?y))))

- (plus some technicalities, optimizations)
- size of PDDL depends on size of Datalog$\neg$ rewritings
**Main Compilability Result**

**Theorem**

For the class of eKAB instances for which all ECQs are Datalog\(^\neg\) rewritable w.r.t. the TBox, there is a compilation scheme into PDDL that preserves plan size exactly.

- use **derived predicates** to include Datalog\(^\neg\) rules in PDDL 2.2:
  
  ```
  RiskOfPhysicalDamage(x) ← Drone(x) ∧ near(x, y) ∧ MovingObject(y)
  (:derived (RiskOfPhysicalDamage ?x)
   (exists ?y (and (Drone ?x) (near ?x ?y) (MovingObject ?y)))
  )
  ```

- (plus some technicalities, optimizations)
- size of PDDL depends on size of Datalog\(^\neg\) rewritings
- immediately applicable to *existing* Datalog\(^\neg\) rewritings for DLs
- [new polynomial-size Datalog\(^\neg\) rewriting for Horn-ALCHOIQ]
Polynomial-size Datalog\textsuperscript{−} rewriting for Horn-\textit{ALCHOIQ}

Combined rewriting for Horn-\textit{ALCHOIQ}: \cite{Carral2018}  

- **exponential** Datalog program using types  
  \[ C(x) \rightarrow R(x, t_D) \land D(t_D) \]

- complex filtration phase  
  “check whether \( F_{q,\sigma} \) is a rooted directed forest”

\[ C(x) \rightarrow R(x, t_D) \land D(t_D) \]
Polynomial-size Datalog\(-\) rewriting for Horn-\textit{ALCHOIQ}

Combined rewriting for Horn-\textit{ALCHOIQ}: \cite{Carral2018}

- exponential Datalog program using types \(C(x) \rightarrow R(x, t_D) \land D(t_D)\)
- complex filtration phase \(\text{“check whether } F_{q,\sigma} \text{ is a rooted directed forest”}\)

Transform into polynomial rule set:

- represent types using Datalog\(^S\) set terms \(\text{concept}(C, X) \rightarrow \text{role}(r, X, \{D\}) \land \text{concept}(D, \{D\})\)

\cite{Ortiz2010}
Polynomial-size Datalog\(^\neg\) rewriting for Horn-\textit{ALCHOIQ}

Combined rewriting for Horn-\textit{ALCHOIQ}:

- exponential Datalog program using types
  \[ C(x) \rightarrow R(x, t_D) \land D(t_D) \]
- complex filtration phase
  “check whether \( F_{q,\sigma} \) is a rooted directed forest”

Transform into polynomial rule set:

- represent types using Datalog\(^S\) set terms
  \[ \text{concept}(C, X) \rightarrow \text{role}(r, X, \{ D \}) \land \text{concept}(D, \{ D \}) \]

- encode filtration into stratified Datalog\(^S,\neg\)
  \[ \text{edge}(i, j, V_1, \ldots, V_k) \rightarrow \text{reach}(i, j, V_1, \ldots, V_k) \]
  \[ \ldots \]
  \[ \text{reach}(i, i, V_1, \ldots, V_k) \rightarrow \text{bad}(V_1, \ldots, V_k) \]
Polynomial-size Datalog\(^-\) rewriting for Horn-\(\text{ALCHOIQ}\)

**Combined rewriting for Horn-\(\text{ALCHOIQ}\):**  
(Carral, Dragoste, and Krötzsch 2018)

- **exponential** Datalog program using types 
  \(C(x) \rightarrow R(x, t_D) \land D(t_D)\)
- **complex filtration phase**  
  “check whether \(F_{q,\sigma}\) is a rooted directed forest”

Transform into polynomial rule set:

- **represent types using Datalog\(^S\) set terms**  
  \(\text{concept}(C, X) \rightarrow \text{role}(r, X, \{D\}) \land \text{concept}(D, \{D\})\)

- **encode filtration into stratified Datalog\(^S,\neg\)**
  \[
  \begin{align*}
  \text{edge}(i, j, V_1, \ldots, V_k) &\rightarrow \text{reach}(i, j, V_1, \ldots, V_k) \\
  \cdots \text{reach}(i, i, V_1, \ldots, V_k) &\rightarrow \text{bad}(V_1, \ldots, V_k)
  \end{align*}
  \]

- **translate set terms back into Datalog\(^-\) using bit vectors**
Non-compilability

(Thiébaux, Hoffmann, and Nebel 2005)

What about more expressive DLs?

Horn-$\text{SROIQ}$ = extension of Horn-$\text{ALCHOIQ}$ with one additional statement type

Theorem

Unless $\text{ExpTime} = \text{NP}$, there is no polynomial-size compilation scheme from Horn-$\text{SROIQ}$ eKABs to PDDL preserving plan size polynomially.

- Based on 2ExpTime-hardness proof for CQ entailment in Horn-$\text{SROIQ}$
- Horn-$\text{SROIQ}$ eKABs (non-uniformly) simulate universal 2ExpTime Turing machine
- Polynomial compilation implies $\text{2ExpTime} \subseteq \text{ExpTime/poly}$
- $\Rightarrow$ weak exponential hierarchy collapses (Buhrman and Homer 1992)
- $\Rightarrow$ any bounded quanti/uniFB01er pre/uniFB01x in second-order logic or Presburger arithmetic could be eliminated (Gottlob, Leone, and Veith 1995; Haase 2014)
- [Similar results for $\text{SH}$, $\text{ALCI}$ eKABs]
Non-compilability

(Thiébaux, Hoffmann, and Nebel 2005)

What about more expressive DLs?

Horn-$SROIQ$ = extension of Horn-$ALCHOIQ$ with one additional statement type

Theorem

Unless $\text{ExpTime}^{NP} = \text{ExpTime}$, there is no polynomial-size compilation scheme from Horn-$SROIQ$ eKABs to PDDL preserving plan size polynomially.

- based on 2ExpTime-hardness proof for CQ entailment in Horn-$SROIQ$
- Horn-$SROIQ$ eKABs (non-uniformly) simulate universal 2ExpTime Turing machine
- polynomial compilation implies $2\text{ExpTime} \subseteq \text{ExpTime/poly}$
Non-compilability

(Thiébaux, Hoffmann, and Nebel 2005)

What about more expressive DLs?

**Horn-SROIQ** = extension of Horn-ALCHOIQ with one additional statement type

---

**Theorem**

*Unless ExpTime\(^{NP} = \) ExpTime, there is no polynomial-size compilation scheme from Horn-SROIQ eKABs to PDDL preserving plan size polynomially.*

- based on 2ExpTime-hardness proof for CQ entailment in Horn-SROIQ
- Horn-SROIQ eKABs (non-uniformly) simulate universal 2ExpTime Turing machine
- polynomial compilation implies 2ExpTime \(\subseteq\) ExpTime/poly
- \(\Rightarrow\) weak exponential hierarchy collapses \(\text{(Buhrman and Homer 1992)}\)
- \(\Rightarrow\) any bounded quantifier prefix in second-order logic or Presburger arithmetic could be eliminated \(\text{(Gottlob, Leone, and Veith 1995; Haase 2014)}\)
Non-compilability

(Thiébaux, Hoffmann, and Nebel 2005)

What about more expressive DLs?

Horn-\textit{SROIQ} = extension of Horn-\textit{ALCHOIQ} with one additional statement type

\textbf{Theorem}

\textit{Unless }\textit{ExpTime}^{NP} = \textit{ExpTime}, there is no polynomial-size compilation scheme from Horn-\textit{SROIQ} eKABs to PDDL preserving plan size polynomially.

- based on 2ExpTime-hardness proof for CQ entailment in Horn-\textit{SROIQ}
- Horn-\textit{SROIQ} eKABs (non-uniformly) simulate universal 2ExpTime Turing machine
- polynomial compilation implies $2\text{ExpTime} \subseteq \text{ExpTime/poly}$
- $\Rightarrow$ weak exponential hierarchy collapses

(Buhrman and Homer 1992)

- $\Rightarrow$ any bounded quantifier prefix in second-order logic or Presburger arithmetic could be eliminated

(Gottlob, Leone, and Veith 1995; Haase 2014)

- [similar results for $\text{SH}$, $\text{ALCI}$ eKABs]
Experiments

- implementation of compilation scheme using practical but exponential Datalog rewritings for Horn-\(SHIQ\) (Clipper) (Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)

- input: .pddl /uniFB01le with mko operators, .ttl/.owl /uniFB01le for the TBox
- output: compiled, normalized standard .pddl /uniFB01le
- planning: FD 20.06 (Helmert 2006)
- benchmarks:
  - 125 existing DL-Lite eKAB instances
  - 110 new, more expressive instances
- comparison with (Calvanese, Montali, Patrizi, and Stawowy 2016; Borgwardt, Hoffmann, Kovtunova, and Steinmetz 2021)
- platform: Intel Core i5-4590 CPU @3.30GHz, 8GB memory, 600s timeout
Experiments

- implementation of compilation scheme using practical but exponential Datalog\^\neg rewriting for Horn-SHIQ (Clipper) (Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)
- input: .pddl file with mko operators, .ttl/.owl file for the TBox
- output: compiled, normalized standard .pddl file
- planning: FD 20.06 (Helmert 2006)
Experiments

- implementation of compilation scheme using practical but exponential Datalog rewritings for Horn-$SHIQ$ (Clipper) (Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)
- input: .pddl file with mko operators, .ttl/.owl file for the TBox
- output: compiled, normalized standard .pddl file
- planning: FD 20.06 (Helmert 2006)
- benchmarks:
  - 125 existing DL-Lite eKAB instances
  - 110 new, more expressive instances
Experiments

- **implementation** of compilation scheme using practical but **exponential** Datalog\(\neg\) rewriting for Horn-\(SHIQ\) (Clipper) (Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)
- **input**: .pddl /uniFB01le with mko operators, .ttl/.owl /uniFB01le for the TBox
- **output**: compiled, **normalized** standard .pddl file
- **planning**: FD 20.06 (Helmert 2006)
- **benchmarks**:
  - 125 existing DL-Lite eKAB instances
  - 110 **new**, more expressive instances
- **comparison** with (Calvanese, Montali, Patrizi, and Stawowy 2016; Borgwardt, Hoffmann, Kovtunova, and Steinmetz 2021)
- **platform**: Intel Core i5-4590 CPU @3.30GHz, 8GB memory, 600s timeout
### Results

<table>
<thead>
<tr>
<th>Domain</th>
<th>#</th>
<th># solved</th>
<th>planning time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Cal16</td>
<td>Bor21</td>
</tr>
<tr>
<td>Cats</td>
<td>20</td>
<td>14</td>
<td>20</td>
</tr>
<tr>
<td>Elevator</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Robot</td>
<td>20</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>TaskAssign</td>
<td>20</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>TPSA</td>
<td>15</td>
<td>14</td>
<td>5</td>
</tr>
<tr>
<td>VTA</td>
<td>15</td>
<td>15</td>
<td>13</td>
</tr>
<tr>
<td>VTA-Roles</td>
<td>15</td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td>∑</td>
<td>125</td>
<td>85</td>
<td>95</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain</th>
<th>#</th>
<th># solved</th>
<th>planning time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drones</td>
<td>24</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Queens</td>
<td>30</td>
<td>15</td>
<td>21.66</td>
</tr>
<tr>
<td>RobotConj</td>
<td>56</td>
<td>56</td>
<td>8.14</td>
</tr>
<tr>
<td>∑</td>
<td>110</td>
<td>91</td>
<td>30.87</td>
</tr>
</tbody>
</table>

- our method ("Horn") solves all DL-Lite instances very fast
- feasible also for more expressive TBoxes
- compilation itself also much faster on average
Summary

- new compilation for eKABs with expressive Horn-DL TBoxes
- outperforms previous eKAB compilations
- theoretical results on polynomial compilability (Horn-\textit{ALCHOIQ} vs. Horn-\textit{SROIQ})
Summary

- new compilation for eKABs with **expressive** Horn-DL TBoxes
- outperforms previous eKAB compilations
- theoretical results on **polynomial compilability** (Horn-ALCHOIQ vs. Horn-SROIQ)

Future work:

- implement practical compilations **beyond** Horn-SHIQ
- adapt to **other semantics** for action effects
Summary

- new compilation for eKABs with expressive Horn-DL TBoxes
- outperforms previous eKAB compilations
- theoretical results on polynomial compilability (Horn-\textsc{ALCHOIQ} vs. Horn-\textsc{SROIQ})

Future work:
- implement practical compilations beyond Horn-\textsc{SHIQ}
- adapt to other semantics for action effects

Thank you!


Picture rights:
2016 Sample Return Robot Challenge (NHQ201609050020) by “NASA HQ PHOTO”, CC BY-NC-ND 2.0
Drone by “ninfaj”, CC BY-NC-ND 2.0
Artist Concept of LCROSS/LRO (NASA, Moon, 6/15/09) by “NASA's Marshall Space Flight Center”, CC BY-NC-ND 2.0