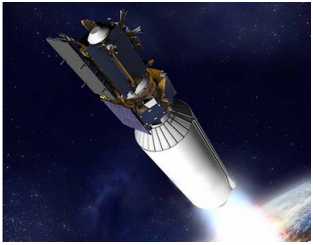
Dresden University of Technology

Stefan Borgwardt
Technische Universität Dresden

Automated Planning with Ontologies

DL Seminar, March 27, 2026

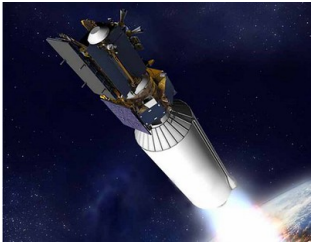
Planning and reasoning



AI planning

- dynamic
- actions modify abstract states
- low-level action knowledge
- fixed domain
- closed-world assumption

Planning and reasoning



AI planning

- dynamic
- actions modify abstract states
- low-level action knowledge
- fixed domain
- closed-world assumption

Knowledge representation and reasoning

- static
- global state constraints
- high-level background knowledge
- open domain
- open-world assumption

Outline

Preliminaries

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

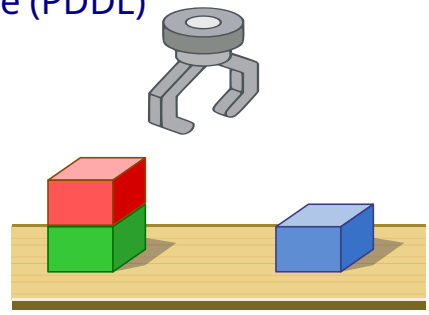
(Calvanese, Montali, Patrizi, and Stawowy 2016)

Optimizing the DL-Lite compilation (B, Hoffmann, Kovtunova, and Steinmetz 2021)

Compilations for Horn DLs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

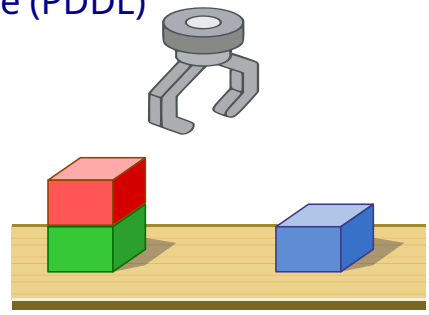
Coherent updates for DL-Lite (B, Nhu, and Röger 2025)

Planning domain definition language (PDDL)

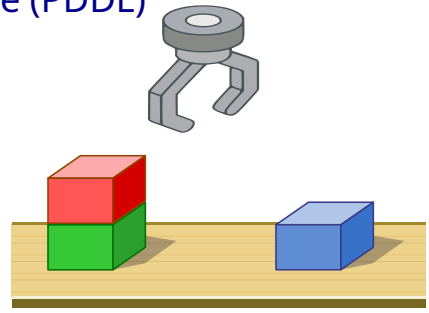


Planning domain definition language (PDDL)

Predicates: `on/2`, `clear/1`, `table/1`



Planning domain definition language (PDDL)



Predicates: `on`/2, `clear`/1, `table`/1

Action: `move`

Parameters: $\langle x, y \rangle$

Precondition: $\texttt{clear}(x) \wedge \texttt{clear}(y) \wedge \neg\texttt{table}(x)$

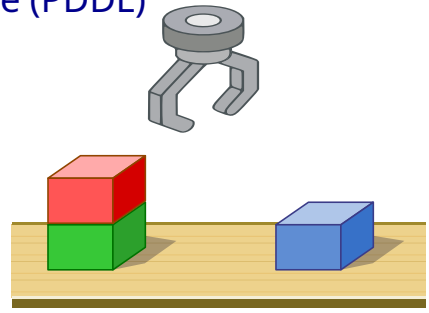
Effects:

$\texttt{on}(x, y)$

If $\neg\texttt{table}(y)$, then $\neg\texttt{clear}(y)$

For all z, if $\texttt{on}(x, z)$, then $\texttt{clear}(z)$ and $\neg\texttt{on}(x, z)$

Planning domain definition language (PDDL)



Predicates: on/2, clear/1, table/1

Action: move

Parameters: $\langle x, y \rangle$

Precondition: $\texttt{clear}(x) \wedge \texttt{clear}(y) \wedge \neg\texttt{table}(x)$

Effects:

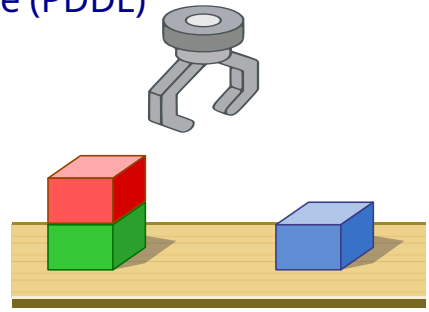
$\texttt{on}(x, y)$

If $\neg\texttt{table}(y)$, then $\neg\texttt{clear}(y)$

For all z, if $\texttt{on}(x, z)$, then $\texttt{clear}(z)$ and $\neg\texttt{on}(x, z)$

Objects: r, g, b, t

Planning domain definition language (PDDL)



Predicates: on/2, clear/1, table/1

Action: move

Parameters: $\langle x, y \rangle$

Precondition: $\texttt{clear}(x) \wedge \texttt{clear}(y) \wedge \neg\texttt{table}(x)$

Effects:

$\texttt{on}(x, y)$

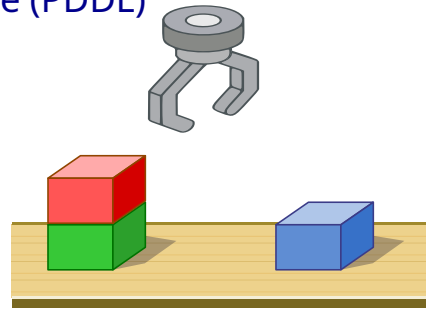
If $\neg\texttt{table}(y)$, then $\neg\texttt{clear}(y)$

For all z, if $\texttt{on}(x, z)$, then $\texttt{clear}(z)$ and $\neg\texttt{on}(x, z)$

Objects: r, g, b, t

Initial state: $\texttt{on}(\texttt{r}, \texttt{g})$, $\texttt{on}(\texttt{g}, \texttt{t})$, $\texttt{on}(\texttt{b}, \texttt{t})$, $\texttt{clear}(\texttt{r})$, $\texttt{clear}(\texttt{b})$, $\texttt{clear}(\texttt{t})$, $\texttt{table}(\texttt{t})$

Planning domain definition language (PDDL)



Predicates: on/2, clear/1, table/1

Action: move

Parameters: $\langle x, y \rangle$

Precondition: $\texttt{clear}(x) \wedge \texttt{clear}(y) \wedge \neg\texttt{table}(x)$

Effects:

$\texttt{on}(x, y)$

If $\neg\texttt{table}(y)$, then $\neg\texttt{clear}(y)$

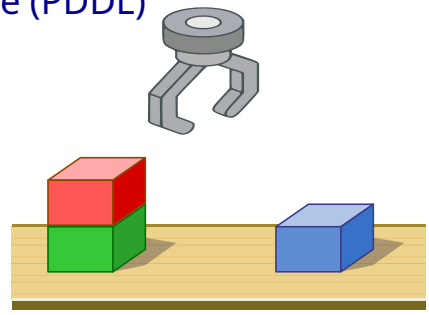
For all z, if $\texttt{on}(x, z)$, then $\texttt{clear}(z)$ and $\neg\texttt{on}(x, z)$

Objects: r, g, b, t

Initial state: $\texttt{on}(\texttt{r}, \texttt{g})$, $\texttt{on}(\texttt{g}, \texttt{t})$, $\texttt{on}(\texttt{b}, \texttt{t})$, $\texttt{clear}(\texttt{r})$, $\texttt{clear}(\texttt{b})$, $\texttt{clear}(\texttt{t})$, $\texttt{table}(\texttt{t})$

Goal formula: $\texttt{on}(\texttt{g}, \texttt{r})$

Planning domain definition language (PDDL)



Predicates: on/2, clear/1, table/1

Action: move

Parameters: $\langle x, y \rangle$

Precondition: $\texttt{clear}(x) \wedge \texttt{clear}(y) \wedge \neg\texttt{table}(x)$

Effects:

$\texttt{on}(x, y)$

If $\neg\texttt{table}(y)$, then $\neg\texttt{clear}(y)$

For all z, if $\texttt{on}(x, z)$, then $\texttt{clear}(z)$ and $\neg\texttt{on}(x, z)$

Objects: r, g, b, t

Initial state: $\texttt{on}(\texttt{r}, \texttt{g})$, $\texttt{on}(\texttt{g}, \texttt{t})$, $\texttt{on}(\texttt{b}, \texttt{t})$, $\texttt{clear}(\texttt{r})$, $\texttt{clear}(\texttt{b})$, $\texttt{clear}(\texttt{t})$, $\texttt{table}(\texttt{t})$

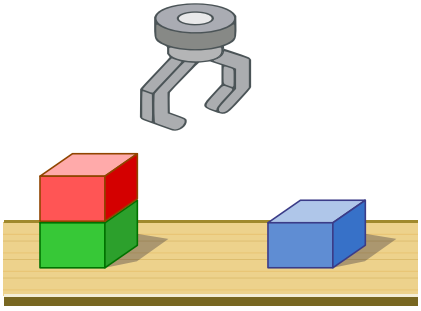
Goal formula: $\texttt{on}(\texttt{g}, \texttt{r})$

PDDL task $(\mathcal{P}, \mathcal{A}, \mathcal{O}, I, G)$

PDDL action effects

Action $(\vec{x}, pre, eff)$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, cond, add, del)$:

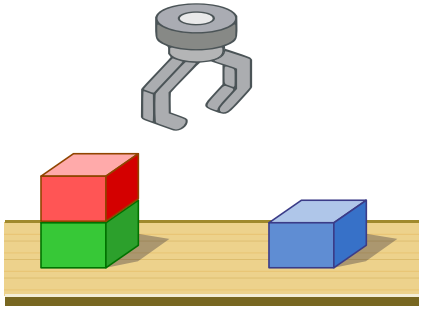


PDDL action effects

Action $(\vec{x}, pre, eff)$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, cond, add, del)$:

(	$\langle\rangle$,	$\top$,	$\{\texttt{on}(x,y)\}$,	$\varnothing$	)
(	$\langle\rangle$,	$\neg\texttt{table}(y)$,	$\varnothing$,	$\{\neg\texttt{clear}(y)\}$	)
(	$\langle z\rangle$,	$\texttt{on}(x,z)$,	$\{\texttt{clear}(z)\}$,	$\{\neg\texttt{on}(x,z)\}$	)

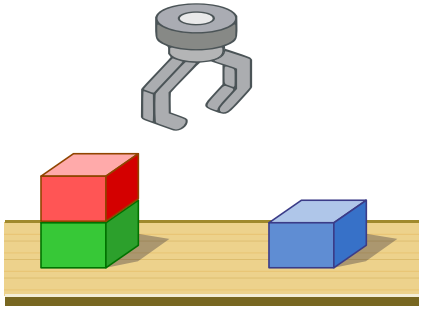


PDDL action effects

Action $(\vec{x}, pre, eff)$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, cond, add, del)$:

(	$\langle\rangle$,	$\top$,	$\{\texttt{on}(x,y)\}$,	$\varnothing$	)
(	$\langle\rangle$,	$\neg\texttt{table}(y)$,	$\varnothing$,	$\{\neg\texttt{clear}(y)\}$	)
(	$\langle z\rangle$,	$\texttt{on}(x,z)$,	$\{\texttt{clear}(z)\}$,	$\{\neg\texttt{on}(x,z)\}$	)



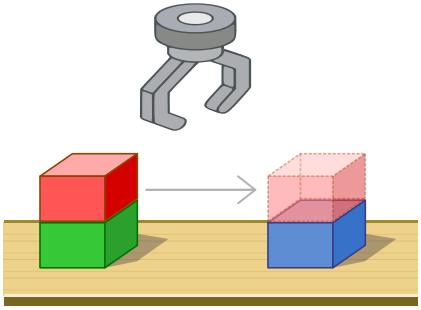
State *s*: set of ground atoms (*facts*)

PDDL action effects

Action $(\vec{x}, pre, eff)$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, cond, add, del)$:

$$
\begin{array}{llll}
(\langle\rangle, & \top, & \{\texttt{on}(x,y)\}, & \varnothing) \\
(\langle\rangle, & \neg\texttt{table}(y), & \varnothing, & \{\neg\texttt{clear}(y)\}) \\
(\langle z\rangle, & \texttt{on}(x,z), & \{\texttt{clear}(z)\}, & \{\neg\texttt{on}(x,z)\})
\end{array}
$$



State *s*: set of ground atoms (*facts*)

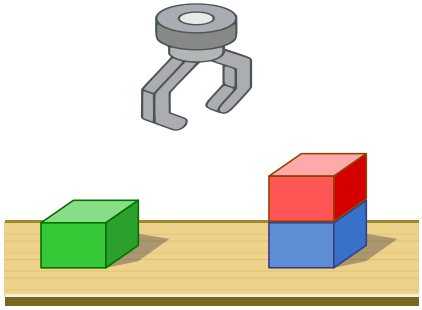
Ground action $\texttt{move}(\texttt{r},\texttt{b})$ is applicable if $s \models \texttt{clear}(\texttt{r}) \wedge \texttt{clear}(\texttt{b}) \wedge \neg\texttt{table}(\texttt{r})$

PDDL action effects

Action $(\vec{x}, \mathit{pre}, \mathit{eff})$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, \mathit{cond}, \mathit{add}, \mathit{del})$:

$$
\begin{array}{llll}
(\langle\rangle, & \top, & \{\texttt{on}(x,y)\}, & \varnothing) \\
(\langle\rangle, & \neg\texttt{table}(y), & \varnothing, & \{\neg\texttt{clear}(y)\}) \\
(\langle z\rangle, & \texttt{on}(x,z), & \{\texttt{clear}(z)\}, & \{\neg\texttt{on}(x,z)\})
\end{array}
$$



State *s*: set of ground atoms (*facts*)

Ground action $\texttt{move}(\texttt{r},\texttt{b})$ is applicable if $s \models \texttt{clear}(\texttt{r}) \wedge \texttt{clear}(\texttt{b}) \wedge \neg\texttt{table}(\texttt{r})$

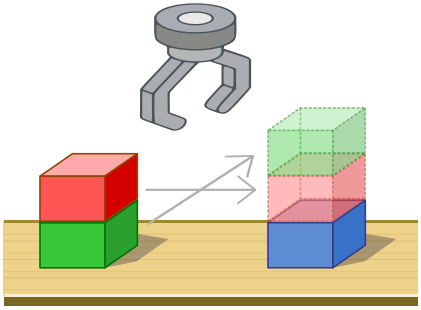
Application yields new state: $\{\ \texttt{on}(\texttt{r},\texttt{b}),\ \texttt{on}(\texttt{g},\texttt{t}),\ \texttt{on}(\texttt{b},\texttt{t}),\ \ldots\ \}$

PDDL action effects

Action $(\vec{x}, pre, eff)$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, cond, add, del)$:

$$
\begin{array}{llll}
(\langle\rangle, & \top, & \{\texttt{on}(x,y)\}, & \varnothing) \\
(\langle\rangle, & \neg\texttt{table}(y), & \varnothing, & \{\neg\texttt{clear}(y)\}) \\
(\langle z\rangle, & \texttt{on}(x,z), & \{\texttt{clear}(z)\}, & \{\neg\texttt{on}(x,z)\})
\end{array}
$$



State *s*: set of ground atoms (*facts*)

Ground action $\texttt{move}(\texttt{r},\texttt{b})$ is applicable if $s \models \texttt{clear}(\texttt{r}) \wedge \texttt{clear}(\texttt{b}) \wedge \neg\texttt{table}(\texttt{r})$

Application yields new state: $\{\ \texttt{on}(\texttt{r},\texttt{b}),\ \texttt{on}(\texttt{g},\texttt{t}),\ \texttt{on}(\texttt{b},\texttt{t}),\ \ldots\ \}$

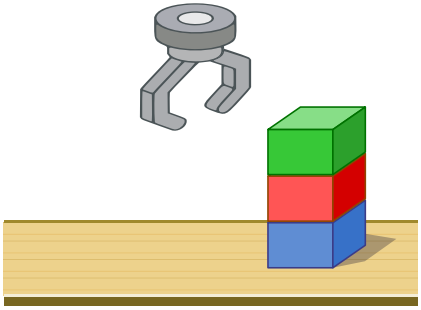
Plan: $\langle \texttt{move}(\texttt{r},\texttt{b}),\ \texttt{move}(\texttt{g},\texttt{r}) \rangle$ reaches the goal $\texttt{on}(\texttt{g},\texttt{r})$

PDDL action effects

Action $(\vec{x}, pre, eff)$:

- parameters $\vec{x}$
- precondition formula *pre*
- conditional effects $(\vec{y}, cond, add, del)$:

$$
\begin{array}{llll}
(\ \langle\rangle, & \top, & \{\texttt{on}(x,y)\}, & \varnothing \ ) \\
(\ \langle\rangle, & \neg\texttt{table}(y), & \varnothing, & \{\neg\texttt{clear}(y)\} \ ) \\
(\ \langle z\rangle, & \texttt{on}(x,z), & \{\texttt{clear}(z)\}, & \{\neg\texttt{on}(x,z)\} \ )
\end{array}
$$



State *s*: set of ground atoms (*facts*)

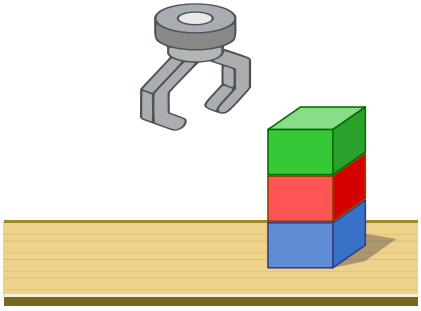
Ground action $\texttt{move}(\texttt{r},\texttt{b})$ is applicable if $s \models \texttt{clear}(\texttt{r}) \wedge \texttt{clear}(\texttt{b}) \wedge \neg\texttt{table}(\texttt{r})$

Application yields new state: $\{\ \texttt{on}(\texttt{r},\texttt{b}),\ \texttt{on}(\texttt{g},\texttt{t}),\ \texttt{on}(\texttt{b},\texttt{t}),\ \ldots\ \}$

Plan: $\langle \texttt{move}(\texttt{r},\texttt{b}),\ \texttt{move}(\texttt{g},\texttt{r})\rangle$ reaches the goal $\texttt{on}(\texttt{g},\texttt{r})$

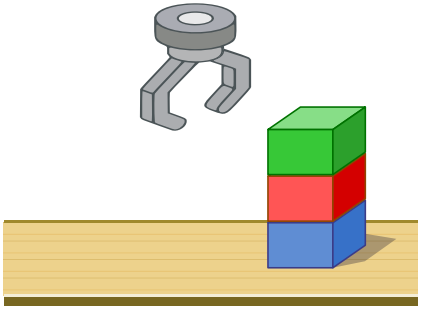
PDDL properties

- many planning benchmarks, competitions
- first-order formulas in conditions and goal
- closed-world, closed-domain states



PDDL properties

- many planning benchmarks, competitions
- first-order formulas in conditions and goal
- closed-world, closed-domain states
- checking conditions is PSPACE-complete
- plan existence is EXPSPACE-complete
- all facts updated explicitly, lots of bookkeeping



(Erol, Nau, and Subrahmanian 1995)

Outline

Preliminaries

- Planning domain definition language (PDDL)
- Description logics (DLs)
- Explicit-input knowledge and action bases (eKABs) (Calvanese, Montali, Patrizi, and Stawowy 2016)

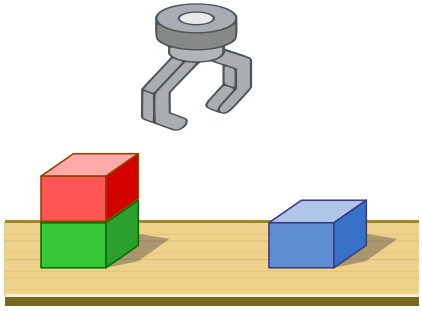
Optimizing the DL-Lite compilation (B, Hoffmann, Kovtunova, and Steinmetz 2021)

Compilations for Horn DLs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Coherent updates for DL-Lite (B, Nhu, and Röger 2025)

Description logics (DLs)

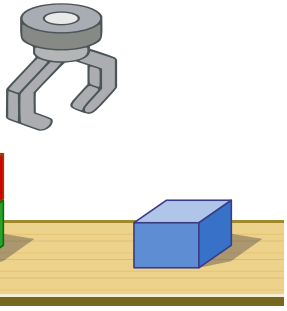
Decidable fragments of FOL with a funny syntax
(and only unary and binary predicates)



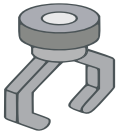
Description logics (DLs)

Decidable fragments of FOL with a funny syntax
(and only unary and binary predicates)

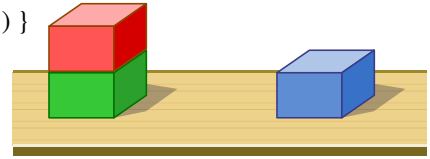
ABox $\mathcal{A} = \{\texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$



Description logics (DLs)



Decidable fragments of FOL with a funny syntax
(and only unary and binary predicates)



ABox $\mathcal{A} = \{\texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$

TBox/ontology $\mathcal{T} = \{$

$\texttt{onblock} \sqsubseteq \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \texttt{funct onblock},$

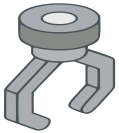
$\texttt{ontable} \sqsubseteq \texttt{on},\ \exists \texttt{ontable}^- \sqsubseteq \texttt{Table},\ \texttt{Block} \sqsubseteq \neg\texttt{Table},$

$\texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Blocked},\ \exists \texttt{onblock} \sqsubseteq \neg \exists \texttt{ontable}$

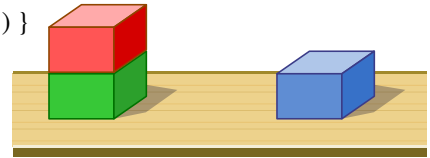
$\texttt{Clear} \equiv \neg\texttt{Blocked},\ \texttt{Block} \sqsubseteq \exists \texttt{onblock} \sqcup \exists \texttt{ontable},\ \texttt{Robot} \sqsubseteq\ = 2\,\texttt{hand}\ \}$

Description logics (DLs)

Decidable fragments of FOL with a funny syntax
(and only unary and binary predicates)



ABox $\mathcal{A} = \{\texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$



TBox/ontology $\mathcal{T} = \{$

$\texttt{onblock} \sqsubseteq \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \texttt{funct onblock},$

$\texttt{ontable} \sqsubseteq \texttt{on},\ \exists \texttt{ontable}^- \sqsubseteq \texttt{Table},\ \texttt{Block} \sqsubseteq \neg\texttt{Table},$

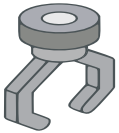
$\texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Blocked},\ \exists \texttt{onblock} \sqsubseteq \neg\exists \texttt{ontable}$

~~$\texttt{Clear} \equiv \neg\texttt{Blocked},\ \texttt{Block} \sqsubseteq \exists \texttt{onblock} \sqcup \exists \texttt{ontable},\ \texttt{Robot} \sqsubseteq = 2\,\texttt{hand}$~~ $\}$

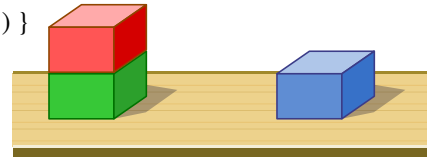
Some (Horn) DLs: DL-Lite, $\mathcal{ELH}_\bot$, Horn-$\mathcal{ALCHOIQ}$, Horn-$\mathcal{SROIQ}$, ~~$\mathcal{ALCI}$, $\mathcal{SH}$~~, ...

Description logics (DLs)

Decidable fragments of FOL with a funny syntax
(and only unary and binary predicates)



ABox $\mathcal{A} = \{\texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$



TBox/ontology $\mathcal{T} = \{$

$\texttt{onblock} \sqsubseteq \texttt{on},\ \exists\texttt{onblock}^- \sqsubseteq \texttt{Block},\ \texttt{funct onblock},$

$\texttt{ontable} \sqsubseteq \texttt{on},\ \exists\texttt{ontable}^- \sqsubseteq \texttt{Table},\ \texttt{Block} \sqsubseteq \neg\texttt{Table},$

$\texttt{Block} \equiv \exists\texttt{on},\ \exists\texttt{onblock}^- \sqsubseteq \texttt{Blocked},\ \exists\texttt{onblock} \sqsubseteq \neg\exists\texttt{ontable}$

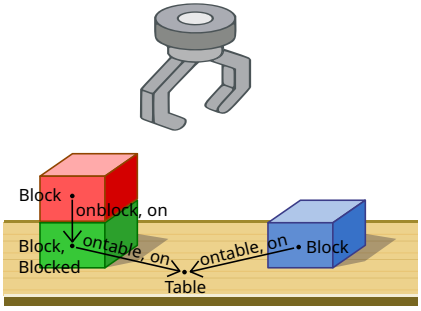
~~$\texttt{Clear} \equiv \neg\texttt{Blocked},\ \texttt{Block} \sqsubseteq \exists\texttt{onblock} \sqcup \exists\texttt{ontable},\ \texttt{Robot} \sqsubseteq = 2\,\texttt{hand}$~~ $\}$

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

Conjunctive queries (CQs): $q(x, z) = \exists y.\ \texttt{on}(x, y) \wedge \texttt{on}(y, z)$

DL semantics

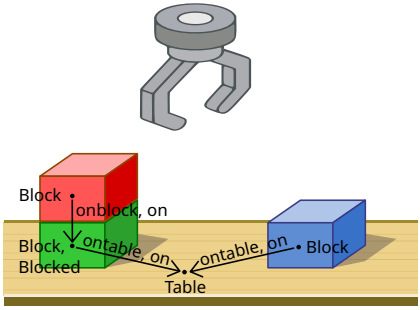
Defined on first-order interpretations $\mathcal{I}$



DL semantics

Defined on first-order interpretations $\mathcal{I}$

- Models: $\mathcal{I} \models \mathcal{T}, \mathcal{I} \models s$
- $s, \mathcal{T}$ are consistent if they have a model
- $s, \mathcal{T} \models q(\vec{c})$ if every model of $s, \mathcal{T}$ satisfies $q(\vec{c})$



DL semantics

Defined on first-order interpretations $\mathcal{I}$

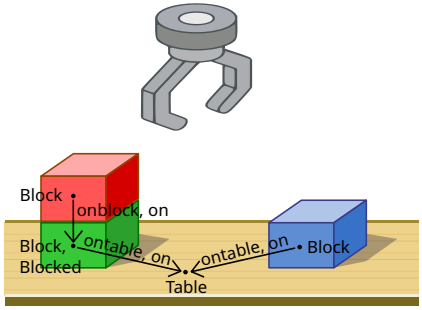
- Models: $\mathcal{I} \models \mathcal{T}, \mathcal{I} \models s$
- $s, \mathcal{T}$ are consistent if they have a model
- $s, \mathcal{T} \models q(\vec{c})$ if every model of $s, \mathcal{T}$ satisfies $q(\vec{c})$

$s = \{\ \texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$

$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \ldots\ \}$

$q(x, z) = \exists y.\ \texttt{on}(x, y) \wedge \texttt{on}(y, z)$

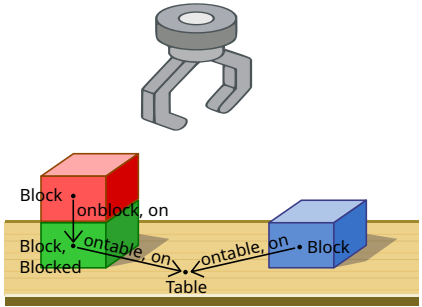
$\rightsquigarrow s, \mathcal{T} \models q(\texttt{r}, \texttt{t})$



DL semantics

Defined on first-order interpretations $\mathcal{I}$

- Models: $\mathcal{I} \models \mathcal{T}, \mathcal{I} \models s$
- $s, \mathcal{T}$ are consistent if they have a model
- $s, \mathcal{T} \models q(\vec{c})$ if every model of $s, \mathcal{T}$ satisfies $q(\vec{c})$



$s = \{\ \texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$

$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \ldots\ \}$

$q(x, z) = \exists y.\ \texttt{on}(x, y) \wedge \texttt{on}(y, z)$

$\rightsquigarrow s, \mathcal{T} \models q(\texttt{r}, \texttt{t})$

- Open-world, open-domain reasoning
- Complexity of CQ entailment ranges from NP to 2ExpTime

Rewriting CQs over DL Ontologies

FO-rewriting: Encode $q(\vec{x})$ and $\mathcal{T}$ into FO-formula $q_{\mathcal{T}}(\vec{x})$

Property: For all ABoxes s, $\underbrace{s, \mathcal{T} \models q(\vec{c})}_{\text{open-world}} \iff \underbrace{\mathcal{I}_s \models q_{\mathcal{T}}(\vec{c})}_{\text{closed-world}}$

Rewriting CQs over DL Ontologies

FO-rewriting: Encode $q(\vec{x})$ and $\mathcal{T}$ into FO-formula $q_{\mathcal{T}}(\vec{x})$

Property: For all ABoxes s, $\underbrace{s, \mathcal{T} \models q(\vec{c})}_{\text{open-world}} \iff \underbrace{\mathcal{I}_s \models q_{\mathcal{T}}(\vec{c})}_{\text{closed-world}}$

$$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^{-} \sqsubseteq \texttt{Block},\ \ldots\ \}$$

Rewriting CQs over DL Ontologies

FO-rewriting: Encode $q(\vec{x})$ and $\mathcal{T}$ into FO-formula $q_{\mathcal{T}}(\vec{x})$

Property: For all ABoxes s, $\underbrace{s, \mathcal{T} \models q(\vec{c})}_{\text{open-world}} \iff \underbrace{\mathcal{I}_s \models q_{\mathcal{T}}(\vec{c})}_{\text{closed-world}}$

$$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \ldots\ \}$$

$$\texttt{Block}(x) \rightsquigarrow \texttt{Block}(x) \vee \exists y.\texttt{onblock}(y,x) \vee \exists y.\texttt{on}(x,y) \vee \exists y.\texttt{onblock}(x,y) \vee \exists y.\texttt{ontable}(x,y)$$

Rewriting CQs over DL Ontologies

FO-rewriting: Encode $q(\vec{x})$ and $\mathcal{T}$ into FO-formula $q_{\mathcal{T}}(\vec{x})$

Property: For all ABoxes s, $\underbrace{s, \mathcal{T} \models q(\vec{c})}_{\text{open-world}} \iff \underbrace{\mathcal{I}_s \models q_{\mathcal{T}}(\vec{c})}_{\text{closed-world}}$

$$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \ldots\ \}$$

$$\texttt{Block}(x) \rightsquigarrow \texttt{Block}(x) \vee \exists y.\texttt{onblock}(y,x) \vee \exists y.\texttt{on}(x,y) \vee \exists y.\texttt{onblock}(x,y) \vee \exists y.\texttt{ontable}(x,y)$$

- Simpler to evaluate (PSPACE, NP)
- Introduced for DL-Lite
- May be very large

(Calvanese, De Giacomo, Lembo, Lenzerini, and Rosati 2007;
Artale, Calvanese, Kontchakov, and Zakharyaschev 2009)

Rewriting CQs over DL Ontolgies into Datalog rules

Datalog-rewriting: For all ABoxes s, $s, \mathcal{T} \models q(\vec{a}) \iff s, \mathcal{R}_{\mathcal{T},q} \models q(\vec{a})$

Rewriting CQs over DL Ontolgies into Datalog rules

Datalog-rewriting: For all ABoxes s, $s, \mathcal{T} \models q(\vec{a}) \iff s, \mathcal{R}_{\mathcal{T},q} \models q(\vec{a})$

$\mathcal{T} = \{ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \dots \}$

$$\texttt{Block}(x) \rightsquigarrow \mathcal{R}_{\mathcal{T},q} = \{ \begin{aligned} q(x) &\leftarrow \texttt{Block}(x), \\ \texttt{Block}(x) &\leftarrow \texttt{onblock}(y, x), \\ \texttt{Block}(x) &\leftarrow \texttt{on}(x, y), \\ \texttt{on}(x, y) &\leftarrow \texttt{onblock}(x, y), \\ \texttt{on}(x, y) &\leftarrow \texttt{ontable}(x, y) \quad \} \end{aligned}$$

Rewriting CQs over DL Ontolgies into Datalog rules

Datalog-rewriting: For all ABoxes $\mathcal{A}$, $\mathcal{A}, \mathcal{T} \models q(\vec{a}) \iff \mathcal{A}, \mathcal{R}_{\mathcal{T},q} \models q(\vec{a})$

$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \ldots\ \}$

$$\texttt{Block}(x) \rightsquigarrow \mathcal{R}_{\mathcal{T},q} = \{ \begin{aligned} q(x) &\leftarrow \texttt{Block}(x), \\ \texttt{Block}(x) &\leftarrow \texttt{onblock}(y, x), \\ \texttt{Block}(x) &\leftarrow \texttt{on}(x, y), \\ \texttt{on}(x, y) &\leftarrow \texttt{onblock}(x, y), \\ \texttt{on}(x, y) &\leftarrow \texttt{ontable}(x, y) \quad \} \end{aligned}$$

- allows recursion
- no existential rules like $\exists y.\texttt{on}(x, y) \leftarrow \texttt{Block}(x)$ ($\texttt{Block} \sqsubseteq \exists \texttt{on}$ in DLs)

Rewriting CQs over DL Ontolgies into Datalog rules

Datalog-rewriting: For all ABoxes $\mathcal{A}$, $\mathcal{A}, \mathcal{T} \models q(\vec{a}) \iff \mathcal{A}, \mathcal{R}_{\mathcal{T},q} \models q(\vec{a})$

$$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^- \sqsubseteq \texttt{Block},\ \ldots\ \}$$

$$\texttt{Block}(x) \rightsquigarrow \mathcal{R}_{\mathcal{T},q} = \{ \begin{aligned} q(x) &\leftarrow \texttt{Block}(x), \\ \texttt{Block}(x) &\leftarrow \texttt{onblock}(y, x), \\ \texttt{Block}(x) &\leftarrow \texttt{on}(x, y), \\ \texttt{on}(x, y) &\leftarrow \texttt{onblock}(x, y), \\ \texttt{on}(x, y) &\leftarrow \texttt{ontable}(x, y) \quad \} \end{aligned}$$

- allows recursion
- no existential rules like $\exists y.\texttt{on}(x, y) \leftarrow \texttt{Block}(x)$ ($\texttt{Block} \sqsubseteq \exists \texttt{on}$ in DLs)
- closed-domain, unique minimal model, ExpTime-complete
- rewritings exist for $\mathcal{ELH}_\bot$ (polynomial size), Horn-$\mathcal{SHIQ}$ (exponential size), ...

(Eiter, Ortiz, Šimkus, Tran, and Xiao 2012; Bienvenu and Ortiz 2015)

Outline

Preliminaries

- Planning domain definition language (PDDL)
- Description logics (DLs)
- Explicit-input knowledge and action bases (eKABs) (Calvanese, Montali, Patrizi, and Stawowy 2016)

Optimizing the DL-Lite compilation (B, Hoffmann, Kovtunova, and Steinmetz 2021)

Compilations for Horn DLs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Coherent updates for DL-Lite (B, Nhu, and Röger 2025)

Explicit-input knowledge and action bases (eKABs)

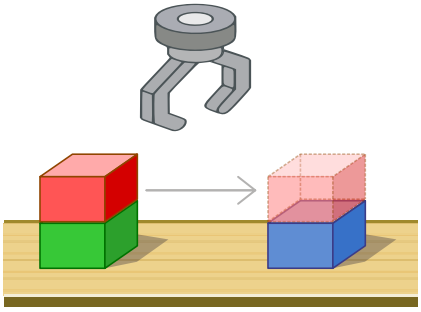
"PDDL + ontology + states=ABoxes + epistemic conjunctive queries (ECQs)"

Explicit-input knowledge and action bases (eKABs)

"PDDL + ontology + states=ABoxes + epistemic conjunctive queries (ECQs)"

$\mathcal{S} = \{\ \mathtt{onblock}(\mathtt{r}, \mathtt{g}),\ \mathtt{ontable}(\mathtt{g}, \mathtt{t}),\ \mathtt{ontable}(\mathtt{b}, \mathtt{t})\ \}$

$\mathcal{T} = \{\ \mathtt{onblock} \sqsubseteq \mathtt{on},\ \mathtt{ontable} \sqsubseteq \mathtt{on},\ \mathtt{Block} \equiv \exists \mathtt{on},\ \exists \mathtt{onblock}^{-} \sqsubseteq \mathtt{Blocked},\ \ldots\ \}$



Explicit-input knowledge and action bases (eKABs)

"PDDL + ontology + states=ABoxes + epistemic conjunctive queries (ECQs)"

$\mathcal{S} = \{\ \texttt{onblock}(\texttt{r}, \texttt{g}),\ \texttt{ontable}(\texttt{g}, \texttt{t}),\ \texttt{ontable}(\texttt{b}, \texttt{t})\ \}$

$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^{-} \sqsubseteq \texttt{Blocked},\ \ldots\ \}$

Action $\texttt{move}(x, y)$:

Precondition:

$\neg[\texttt{Blocked}(x)] \wedge \neg[\texttt{Blocked}(y)] \wedge [\texttt{Block}(x)]$

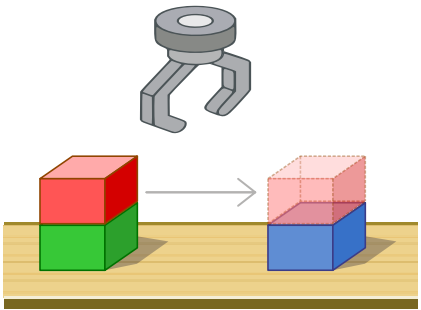
Effects:

If $[\texttt{Table}(y)]$, then $\texttt{ontable}(x, y)$

If $[\texttt{Block}(y)]$, then $\texttt{onblock}(x, y)$

For all z, if $\texttt{ontable}(x, z)$, then $\neg\texttt{ontable}(x, z)$

For all z, if $\texttt{onblock}(x, z)$, then $\neg\texttt{onblock}(x, z)$



Explicit-input knowledge and action bases (eKABs)

eKAB task: PDDL taks $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$ with ontology $\mathcal{T}$ over $\mathcal{P}$

Explicit-input knowledge and action bases (eKABs)

eKAB task: PDDL taks $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$ with ontology $\mathcal{T}$ over $\mathcal{P}$

- Conditions in $\mathcal{A}$ and G are ECQs:

  $$Q ::= p(\vec{x}) \mid [q(\vec{x})] \mid \neg Q \mid Q_1 \wedge Q_2 \mid \exists y.Q$$

  Combination of open-world CQs and closed-world FOL

Explicit-input knowledge and action bases (eKABs)

eKAB task: PDDL taks $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$ with ontology $\mathcal{T}$ over $\mathcal{P}$

- Conditions in $\mathcal{A}$ and G are ECQs:

$$Q ::= p(\vec{x}) \mid [q(\vec{x})] \mid \neg Q \mid Q_1 \wedge Q_2 \mid \exists y.Q$$

Combination of open-world CQs and closed-world FOL

$$\begin{array}{lll} s, \mathcal{T} \models p(\vec{a}) & \text{iff} & s \models p(\vec{a}) \\ s, \mathcal{T} \models [q(\vec{a})] & \text{iff} & s, \mathcal{T} \models q(\vec{a}) \\ s, \mathcal{T} \models \neg Q & \text{iff} & s, \mathcal{T} \not\models Q \\ s, \mathcal{T} \models Q_1 \wedge Q_2 & \text{iff} & s, \mathcal{T} \models Q_1 \text{ and } s, \mathcal{T} \models Q_2 \\ s, \mathcal{T} \models \exists y.Q & \text{iff} & \exists o \in \mathcal{O} : s, \mathcal{T} \models Q[y \mapsto o] \end{array}$$

Explicit-input knowledge and action bases (eKABs)

eKAB task: PDDL taks $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$ with ontology $\mathcal{T}$ over $\mathcal{P}$

- Conditions in $\mathcal{A}$ and G are ECQs:

  $$Q ::= p(\vec{x}) \mid [q(\vec{x})] \mid \neg Q \mid Q_1 \wedge Q_2 \mid \exists y.Q$$

  Combination of open-world CQs and closed-world FOL

  $$\begin{array}{lll} s, \mathcal{T} \models p(\vec{a}) & \text{iff} & s \models p(\vec{a}) \\ s, \mathcal{T} \models [q(\vec{a})] & \text{iff} & s, \mathcal{T} \models q(\vec{a}) \\ s, \mathcal{T} \models \neg Q & \text{iff} & s, \mathcal{T} \not\models Q \\ s, \mathcal{T} \models Q_1 \wedge Q_2 & \text{iff} & s, \mathcal{T} \models Q_1 \text{ and } s, \mathcal{T} \models Q_2 \\ s, \mathcal{T} \models \exists y.Q & \text{iff} & \exists o \in \mathcal{O} : s, \mathcal{T} \models Q[y \mapsto o] \end{array}$$

- Ground action a is applicable if $s, \mathcal{T} \models \mathit{pre}$ and resulting state is consistent with $\mathcal{T}$
- add/del effects operate directly on ABoxes (states)

Explicit-input knowledge and action bases (eKABs)

eKAB task: PDDL taks $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$ with ontology $\mathcal{T}$ over $\mathcal{P}$

- Conditions in $\mathcal{A}$ and G are ECQs:

  $$Q ::= p(\vec{x}) \mid [q(\vec{x})] \mid \neg Q \mid Q_1 \wedge Q_2 \mid \exists y.Q$$

  Combination of open-world CQs and closed-world FOL

  $$\begin{array}{lll} s, \mathcal{T} \models p(\vec{a}) & \text{iff} & s \models p(\vec{a}) \\ s, \mathcal{T} \models [q(\vec{a})] & \text{iff} & s, \mathcal{T} \models q(\vec{a}) \\ s, \mathcal{T} \models \neg Q & \text{iff} & s, \mathcal{T} \not\models Q \\ s, \mathcal{T} \models Q_1 \wedge Q_2 & \text{iff} & s, \mathcal{T} \models Q_1 \text{ and } s, \mathcal{T} \models Q_2 \\ s, \mathcal{T} \models \exists y.Q & \text{iff} & \exists o \in \mathcal{O} : s, \mathcal{T} \models Q[y \mapsto o] \end{array}$$

- Ground action a is applicable if $s, \mathcal{T} \models \mathit{pre}$ and resulting state is consistent with $\mathcal{T}$
- add/del effects operate directly on ABoxes (states)

- separate static part (ontology) and dynamic part (actions)
- epistemic evaluation of ECQ conditions depends on the ontology language (PSPACE^C)

Compiling eKABs into PDDL

DL-Lite: use FO-rewriting to replace CQs by FO formulas

$[\texttt{Block}(x)] \rightsquigarrow$
$\texttt{Block}(x) \vee \exists y.\texttt{onblock}(y,x) \vee \exists y.\texttt{on}(x,y) \vee \exists y.\texttt{onblock}(x,y) \vee \exists y.\texttt{ontable}(x,y)$

Compiling eKABs into PDDL

DL-Lite: use FO-rewriting to replace CQs by FO formulas

$[\texttt{Block}(x)] \rightsquigarrow$
$\texttt{Block}(x) \vee \exists y.\texttt{onblock}(y,x) \vee \exists y.\texttt{on}(x,y) \vee \exists y.\texttt{onblock}(x,y) \vee \exists y.\texttt{ontable}(x,y)$

This yields a compilation scheme: (Nebel 2000; Thiébaux, Hoffmann, and Nebel 2005)

- mappings f_a, f_i, f_g
- $\underbrace{(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)}_{\text{eKAB task}}$ has a plan $\iff$ $\underbrace{(\mathcal{P}, f_a(\mathcal{A}, \mathcal{T}), \mathcal{O}, f_i(I), f_g(G))}_{\text{PDDL task}}$ has a plan

Compiling eKABs into PDDL

DL-Lite: use FO-rewriting to replace CQs by FO formulas

$[\texttt{Block}(x)] \rightsquigarrow$
$\texttt{Block}(x) \vee \exists y.\texttt{onblock}(y,x) \vee \exists y.\texttt{on}(x,y) \vee \exists y.\texttt{onblock}(x,y) \vee \exists y.\texttt{ontable}(x,y)$

This yields a compilation scheme: (Nebel 2000; Thiébaux, Hoffmann, and Nebel 2005)

- mappings f_a, f_i, f_g
- $\underbrace{(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)}_{\text{eKAB task}}$ has a plan $\iff$ $\underbrace{(\mathcal{P}, f_a(\mathcal{A}, \mathcal{T}), \mathcal{O}, f_i(I), f_g(G))}_{\text{PDDL task}}$ has a plan

Theorem (cf. Calvanese, Montali, Patrizi, and Stawowy 2016)

There is an exponential-size compilation scheme from DL-Lite eKABs into PDDL that preserves plan length.

Outline

Preliminaries

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

(Calvanese, Montali, Patrizi, and Stawowy 2016)

Optimizing the DL-Lite compilation (B, Hoffmann, Kovtunova, and Steinmetz 2021)

Compilations for Horn DLs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Coherent updates for DL-Lite (B, Nhu, and Röger 2025)

Optimizing the DL-Lite compilation

FO-rewritings are not well-suited for preprocessing by off-the-shelf planners

- rewritten conditions are grounded and converted to DNF

Optimizing the DL-Lite compilation

FO-rewritings are not well-suited for preprocessing by off-the-shelf planners

- rewritten conditions are grounded and converted to DNF

compile $\rightsquigarrow$
$x \mapsto \mathtt{r}$, ground $\rightsquigarrow$
DNF $\rightsquigarrow$

$$\neg[\mathtt{Blocked}(x)]$$
$$\neg\big(\mathtt{Blocked}(x) \vee \exists y.\mathtt{onblock}(y, x)\big)$$
$$\neg\big(\mathtt{Blocked}(\mathtt{r}) \vee \mathtt{onblock}(\mathtt{r}, \mathtt{r}) \vee \mathtt{onblock}(\mathtt{g}, \mathtt{r}) \vee \mathtt{onblock}(\mathtt{b}, \mathtt{r}) \vee \mathtt{onblock}(\mathtt{t}, \mathtt{r})\big)$$
$$\neg\mathtt{Blocked}(\mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{r}, \mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{g}, \mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{b}, \mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{t}, \mathtt{r})$$

Optimizing the DL-Lite compilation

FO-rewritings are not well-suited for preprocessing by off-the-shelf planners

- rewritten conditions are grounded and converted to DNF

compile $\rightsquigarrow$
$x \mapsto \mathtt{r}$, ground $\rightsquigarrow$
DNF $\rightsquigarrow$

$$\neg[\mathtt{Blocked}(x)]$$
$$\neg\big(\mathtt{Blocked}(x) \vee \exists y.\mathtt{onblock}(y, x)\big)$$
$$\neg\big(\mathtt{Blocked}(\mathtt{r}) \vee \mathtt{onblock}(\mathtt{r}, \mathtt{r}) \vee \mathtt{onblock}(\mathtt{g}, \mathtt{r}) \vee \mathtt{onblock}(\mathtt{b}, \mathtt{r}) \vee \mathtt{onblock}(\mathtt{t}, \mathtt{r})\big)$$
$$\neg\mathtt{Blocked}(\mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{r}, \mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{g}, \mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{b}, \mathtt{r}) \wedge \neg\mathtt{onblock}(\mathtt{t}, \mathtt{r})$$

- $[q(\vec{x})]$ can result in large DNF
- DNF for $\neg[q(\vec{x})]$ exponentially larger

Derived predicates

PDDL 2.2 to the rescue!

- adds derived predicates: set $\mathcal{R}$ of first-order rules $P(\vec{x}) \leftarrow \varphi(\vec{x})$

Derived predicates

PDDL 2.2 to the rescue!

- adds derived predicates: set $\mathcal{R}$ of first-order rules $P(\vec{x}) \leftarrow \varphi(\vec{x})$
- planning semantics: $s \models \mathit{pre}/\mathit{cond}/G$ replaced by $s, \mathcal{R} \models \mathit{pre}/\mathit{cond}/G$
- derived predicate P cannot occur in *add* or *del* effects

Derived predicates

PDDL 2.2 to the rescue!

- adds derived predicates: set $\mathcal{R}$ of first-order rules $P(\vec{x}) \leftarrow \varphi(\vec{x})$
- planning semantics: $s \models pre/cond/G$ replaced by $s, \mathcal{R} \models pre/cond/G$
- derived predicate P cannot occur in *add* or *del* effects

We can flatten complex conditions by auxiliary predicates:

$$\neg[\texttt{Blocked}(x)]$$

$$\overset{\text{compile}}{\rightsquigarrow} \quad \neg\big(\texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)\big)$$

$$\overset{\text{replace}}{\rightsquigarrow} \quad \neg\texttt{Aux}(x) \quad \text{with rule} \quad \texttt{Aux}(x) \leftarrow \texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)$$

Derived predicates

PDDL 2.2 to the rescue!

- adds derived predicates: set $\mathcal{R}$ of first-order rules $P(\vec{x}) \leftarrow \varphi(\vec{x})$
- planning semantics: $s \models \mathit{pre}/\mathit{cond}/G$ replaced by $s, \mathcal{R} \models \mathit{pre}/\mathit{cond}/G$
- derived predicate P cannot occur in *add* or *del* effects

We can flatten complex conditions by auxiliary predicates:

$$\neg[\texttt{Blocked}(x)]$$

$$\overset{\text{compile}}{\rightsquigarrow} \quad \neg\big(\texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)\big)$$

$$\overset{\text{replace}}{\rightsquigarrow} \quad \neg\texttt{Aux}(x) \quad \text{with rule} \quad \texttt{Aux}(x) \leftarrow \texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)$$

- $\mathcal{R}$ needs to be stratified (no recursion through negation)
- still closed-domain, unique minimal model, ExpTime-complete

Derived predicates

PDDL 2.2 to the rescue!

- adds derived predicates: set $\mathcal{R}$ of first-order rules $P(\vec{x}) \leftarrow \varphi(\vec{x})$
- planning semantics: $s \models \mathit{pre}/\mathit{cond}/G$ replaced by $s, \mathcal{R} \models \mathit{pre}/\mathit{cond}/G$
- derived predicate P cannot occur in *add* or *del* effects

We can flatten complex conditions by auxiliary predicates:

$$\neg[\texttt{Blocked}(x)]$$

$$\overset{\text{compile}}{\rightsquigarrow} \quad \neg\big(\texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)\big)$$

$$\overset{\text{replace}}{\rightsquigarrow} \quad \neg\texttt{Aux}(x) \quad \text{with rule} \quad \texttt{Aux}(x) \leftarrow \texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)$$

- $\mathcal{R}$ needs to be stratified (no recursion through negation)
- still closed-domain, unique minimal model, ExpTime-complete
- more expressive than stratified Datalog$^\neg$ (Röger and Grundke 2024)
- plan existence remains ExpSpace-complete

Compilation with derived predicates

$\neg[\texttt{Blocked}(x)]$

compile $\rightsquigarrow$ $\neg\big(\texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)\big)$

replace $\rightsquigarrow$ $\neg\texttt{Aux}(x)$ with rule $\texttt{Aux}(x) \leftarrow \texttt{Blocked}(x) \vee \exists y.\texttt{onblock}(y,x)$

Theorem (B, Hoffmann, Kovtunova, and Steinmetz 2021)

There is an exponential-size compilation scheme from DL-Lite eKABs into PDDL with derived predicates that preserves plan length.

KR'21 experiments

- compare to original compilation (Calvanese, Montali, Patrizi, and Stawowy 2016)
- planning: FD 20.06 (Helmert 2006)

KR'21 experiments

- compare to original compilation (Calvanese, Montali, Patrizi, and Stawowy 2016)
- planning: FD 20.06 (Helmert 2006)
- benchmarks:
  - 125 DL-Lite eKAB tasks from different sources (Hoffmann, Weber, Scicluna, Kacmarek, and Ankolekar 2008; Calvanese, Montali, Patrizi, and Stawowy 2016)
  - 80 classical planning tasks with complex conditions

KR'21 results

Domain	#	# solved Orig	# solved KR21	preprocessing time (s) Orig	preprocessing time (s) KR21
Cats	20	14	20	68.34	0.14
Elevator	20	20	20	0.35	0.29
Robot	20	4	20	15.03	10.09
TaskAssign	20	3	20	0.81	0.12
TPSA	15	14	5	1.42	1.83
VTA	15	15	13	1.32	304.23
VTA-Roles	15	15	5	2.25	11.61
Assembly	30	30	30	0.22	0.55
GridPlacement	20	6	20	49.80	7.01
Miconic	30	9	9	56.61	0.33
Overall	**205**	**130**	**162**	**28.23**	**18.61**

Overall better, though not in all domains

Also useful for classical planning tasks

Outline

Preliminaries

- Planning domain definition language (PDDL)
- Description logics (DLs)
- Explicit-input knowledge and action bases (eKABs) (Calvanese, Montali, Patrizi, and Stawowy 2016)

Optimizing the DL-Lite compilation (B, Hoffmann, Kovtunova, and Steinmetz 2021)

Compilations for Horn DLs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Coherent updates for DL-Lite (B, Nhu, and Röger 2025)

Compiling Horn DL eKABs into PDDL

"(stratified) Datalog$^{\neg}$ rewriting $\subseteq$ PDDL derived predicates"

Compiling Horn DL eKABs into PDDL

"(stratified) Datalog$^{\neg}$ rewriting $\subseteq$ PDDL derived predicates"

$\neg[\texttt{Block}(x)] \overset{\text{compile}}{\rightsquigarrow} \neg q(x)$ with rules

$$
\begin{aligned}
q(x) &\leftarrow \texttt{Block}(x) \\
\texttt{Block}(x) &\leftarrow \texttt{onblock}(y, x) \\
\texttt{Block}(x) &\leftarrow \texttt{on}(x, y) \\
\texttt{on}(x, y) &\leftarrow \texttt{onblock}(x, y) \\
\texttt{on}(x, y) &\leftarrow \texttt{ontable}(x, y)
\end{aligned}
$$

Compiling Horn DL eKABs into PDDL

"(stratified) Datalog$^{\neg}$ rewriting $\subseteq$ PDDL derived predicates"

$\neg[\texttt{Block}(x)] \overset{\text{compile}}{\rightsquigarrow} \neg q(x)$ with rules

$$
\begin{aligned}
q(x) &\leftarrow \texttt{Block}(x) \\
\texttt{Block}(x) &\leftarrow \texttt{onblock}(y, x) \\
\texttt{Block}(x) &\leftarrow \texttt{on}(x, y) \\
\texttt{on}(x, y) &\leftarrow \texttt{onblock}(x, y) \\
\texttt{on}(x, y) &\leftarrow \texttt{ontable}(x, y)
\end{aligned}
$$

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

There is a compilation scheme from Datalog$^{\neg}$ rewritable eKABs into PDDL with derived predicates that preserves plan length.

Compilation size

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

There is a compilation scheme from Datalog$^\neg$ rewritable eKABs into PDDL with derived predicates that preserves plan length.

Corollary

There is a polynomial-size compilation scheme from $\mathcal{ELH}_\bot$ eKABs into PDDL with derived predicates that preserves plan length.

(Bienvenu and Ortiz 2015)

Corollary

There is an exponential-size compilation scheme from Horn-$\mathcal{SHIQ}$ eKABs into PDDL with derived predicates that preserves plan length.

(Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)

AAAI'22 experiments

- compilation scheme using Datalog rewriting for Horn-$\mathcal{SHIQ}$ (Clipper)
  (Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)
- compare to (Calvanese, Montali, Patrizi, and Stawowy 2016; B, Hoffmann, Kovtunova, and Steinmetz 2021)

AAAI'22 experiments

- compilation scheme using Datalog rewriting for Horn-$\mathcal{SHIQ}$ (Clipper)
  (Eiter, Ortiz, Šimkus, Tran, and Xiao 2012)
- compare to (Calvanese, Montali, Patrizi, and Stawowy 2016; B, Hoffmann, Kovtunova, and Steinmetz 2021)
- benchmarks:
  - 125 previous DL-Lite eKAB tasks
  - 110 new, more expressive tasks

AAAI'22 results

Domain	#	# solved			planning time (s)		
		Cal16	KR21	AAAI22	Cal16	KR21	AAAI22
Cats	20	14	20	20	63.46	0.13	0.03
Elevator	20	20	20	20	0.36	0.30	0.03
Robot	20	4	12	20	15.05	10.10	0.11
TaskAssign	20	3	20	20	0.81	0.12	0.06
TPSA	15	14	5	15	2.01	2.42	0.30
VTA	15	15	13	15	23.06	371.56	16.91
VTA-Roles	15	15	5	15	2.25	11.61	1.36
Overall	**125**	**85**	**95**	**125**	**19.99**	**77.33**	**3.59**
Drones	24			20			101.42
Queens	30			15			21.66
RobotConj	56			56			8.14
Overall	**110**			**91**			**30.87**

Solves all DL-Lite tasks very quickly

Compilation itself also much faster on average

Feasible also for more expressive ontologies

Horn-$\mathcal{ALCHOIQ}$

What about more expressive DLs?

Horn-$\mathcal{ALCHOIQ}$

What about more expressive DLs?

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

- exponential Datalog program using types $R(x, t_D) \wedge D(t_D) \leftarrow C(x)$
- complex filtration phase "check whether $F_{q,\sigma}$ is a rooted directed forest"

Horn-$\mathcal{ALCHOIQ}$

What about more expressive DLs?

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

- exponential Datalog program using types $R(x, t_D) \wedge D(t_D) \leftarrow C(x)$
- complex filtration phase "check whether $F_{q,\sigma}$ is a rooted directed forest"

Transform into polynomial set of stratified Datalog$^{\neg}$ rules:

- represent types using Datalog^S set terms (Ortiz, Rudolph, and Šimkus 2010)
  $\text{role}(r, X, \{D\}) \wedge \text{concept}(D, \{D\}) \leftarrow \text{concept}(C, X)$

Horn-$\mathcal{ALCHOIQ}$

What about more expressive DLs?

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

- exponential Datalog program using types $R(x, t_D) \wedge D(t_D) \leftarrow C(x)$
- complex filtration phase "check whether $F_{q,\sigma}$ is a rooted directed forest"

Transform into polynomial set of stratified Datalog$^{\neg}$ rules:

- represent types using Datalog^{S} set terms (Ortiz, Rudolph, and Šimkus 2010)
  $\text{role}(r, X, \{D\}) \wedge \text{concept}(D, \{D\}) \leftarrow \text{concept}(C, X)$
- encode filtration into stratified Datalog$^{S,\neg}$
- eliminate set terms using bit vectors

Horn-$\mathcal{ALCHOIQ}$

What about more expressive DLs?

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

- exponential Datalog program using types $R(x, t_D) \wedge D(t_D) \leftarrow C(x)$
- complex filtration phase "check whether $F_{q,\sigma}$ is a rooted directed forest"

Transform into polynomial set of stratified Datalog$^\neg$ rules:

- represent types using Datalog^S set terms (Ortiz, Rudolph, and Šimkus 2010)
  $\text{role}(r, X, \{D\}) \wedge \text{concept}(D, \{D\}) \leftarrow \text{concept}(C, X)$
- encode filtration into stratified Datalog$^{S,\neg}$
- eliminate set terms using bit vectors

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

There is a polynomial-size compilation scheme from Horn-$\mathcal{ALCHOIQ}$ eKABs into PDDL with derived predicates that preserves plan length.

Horn-$\mathcal{SROIQ}$

What about more expressive DLs?

Horn-$\mathcal{SROIQ}$

What about more expressive DLs?

Horn-$\mathcal{SROIQ}$ = extension of Horn-$\mathcal{ALCHOIQ}$ with complex role inclusions

Horn-$\mathcal{SROIQ}$

What about more expressive DLs?

Horn-$\mathcal{SROIQ}$ = extension of Horn-$\mathcal{ALCHOIQ}$ with complex role inclusions

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Unless ExpTime^{NP} = ExpTime, there can be no polynomial-size compilation scheme from Horn-$\mathcal{SROIQ}$ eKABs to PDDL with derived predicates that preserves plan length polynomially.

Horn-$\mathcal{SROIQ}$

What about more expressive DLs?

Horn-$\mathcal{SROIQ}$ = extension of Horn-$\mathcal{ALCHOIQ}$ with complex role inclusions

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Unless ExpTime^{NP} = ExpTime, there can be no polynomial-size compilation scheme from Horn-$\mathcal{SROIQ}$ eKABs to PDDL with derived predicates that preserves plan length polynomially.

- CQ entailment for Horn-$\mathcal{SROIQ}$ is 2ExpTime-hard (Ortiz, Rudolph, and Šimkus 2010)
- Horn-$\mathcal{SROIQ}$ eKABs (non-uniformly) simulate universal 2ExpTime Turing machine

Horn-$\mathcal{SROIQ}$

What about more expressive DLs?

Horn-$\mathcal{SROIQ}$ = extension of Horn-$\mathcal{ALCHOIQ}$ with complex role inclusions

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Unless ExpTime^{NP} = ExpTime, there can be no polynomial-size compilation scheme from Horn-$\mathcal{SROIQ}$ eKABs to PDDL with derived predicates that preserves plan length polynomially.

- CQ entailment for Horn-$\mathcal{SROIQ}$ is 2ExpTime-hard (Ortiz, Rudolph, and Šimkus 2010)
- Horn-$\mathcal{SROIQ}$ eKABs (non-uniformly) simulate universal 2ExpTime Turing machine
- polynomial-size compilation into PDDL with polynomial plan length (ExpTime)
  ⇒ 2ExpTime ⊆ ExpTime/poly
  ⇒ weak exponential hierarchy collapses (Buhrman and Homer 1992)

Non-Horn DLs?

Similar arguments work for $\mathcal{SH}$, $\mathcal{ALCI}$ eKABs (CQ entailment also 2ExpTime-hard)
(Lutz 2008; Eiter, Lutz, Ortiz, and Šimkus 2009)

Theorem (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Unless ExpTime^{NP} = ExpTime, there is no polynomial-size compilation scheme from Horn-$\mathcal{SROIQ}$ / $\mathcal{SH}$ / $\mathcal{ALCI}$ eKABs to PDDL with derived predicates that preserves plan length polynomially.

Outline

Preliminaries

- Planning domain definition language (PDDL)
- Description logics (DLs)
- Explicit-input knowledge and action bases (eKABs) (Calvanese, Montali, Patrizi, and Stawowy 2016)

Optimizing the DL-Lite compilation (B, Hoffmann, Kovtunova, and Steinmetz 2021)

Compilations for Horn DLs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

Coherent updates for DL-Lite (B, Nhu, and Röger 2025)

Shortcomings of eKABs (I)

Action effects ignore the ontology

$\mathcal{T} = \{$ funct onblock, $\exists$onblock $\sqsubseteq \neg\exists$ontable, … $\}$

Action move(x, y):

Precondition:

$\neg$[Blocked(x)] $\wedge$ $\neg$[Blocked(y)] $\wedge$ [Block(x)]

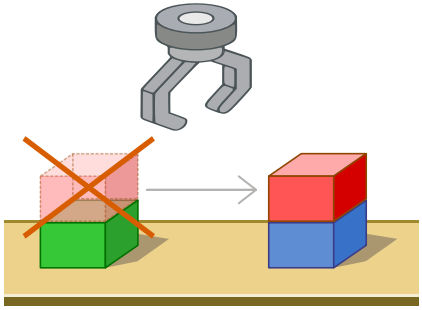
Effects:

If [Table(y)], then ontable(x, y)

If [Block(y)], then onblock(x, y)

For all z, if ontable(x, z), then $\neg$ontable(x, z)

For all z, if onblock(x, z), then $\neg$onblock(x, z)



Shortcomings of eKABs (I)

Action effects ignore the ontology

$\mathcal{T} = \{$ `funct onblock`, $\exists$`onblock` $\sqsubseteq \neg\exists$`ontable`, … $\}$

Action `move`(x, y):

Precondition:

$\neg$[`Blocked`(x)] $\wedge$ $\neg$[`Blocked`(y)] $\wedge$ [`Block`(x)]

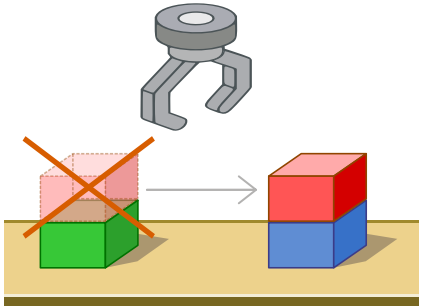
Effects:

If [`Table`(y)], then `ontable`(x, y)

If [`Block`(y)], then `onblock`(x, y)

For all z, if `ontable`(x, z), then $\neg$`ontable`(x, z)

For all z, if `onblock`(x, z), then $\neg$`onblock`(x, z)



Problem I

Old, conflicting `ontable`/`onblock` facts are not removed automatically

Shortcomings of eKABs (II)

Action effects ignore the ontology

$\mathcal{T} = \{\ \texttt{onblock} \sqsubseteq \texttt{on},\ \texttt{ontable} \sqsubseteq \texttt{on},\ \ldots\ \}$

Action $\texttt{move}(x, y)$:

Precondition:

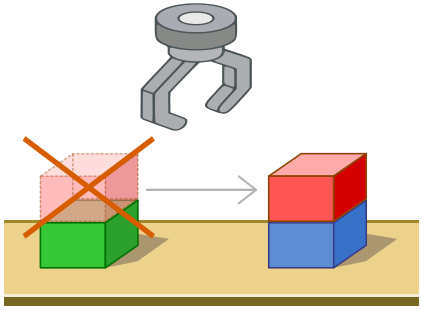
$\neg[\texttt{Blocked}(x)] \wedge \neg[\texttt{Blocked}(y)] \wedge [\texttt{Block}(x)]$

Effects:

If $[\texttt{Table}(y)]$, then $\texttt{ontable}(x, y)$

If $[\texttt{Block}(y)]$, then $\texttt{onblock}(x, y)$

For all z, if $[\texttt{on}(x, z)]$, then $\neg\texttt{on}(x, z)$



Shortcomings of eKABs (II)

Action effects ignore the ontology

$\mathcal{T} = \{$ `onblock` $\sqsubseteq$ `on`, `ontable` $\sqsubseteq$ `on`, … $\}$

Action `move`(x, y):

Precondition:

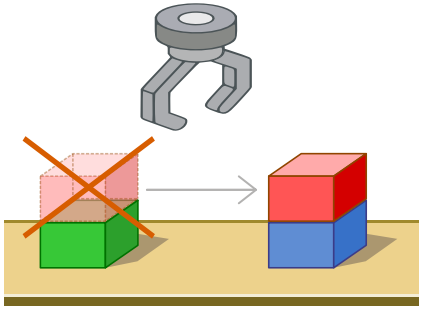
$\neg$[`Blocked`(x)] $\wedge$ $\neg$[`Blocked`(y)] $\wedge$ [`Block`(x)]

Effects:

If [`Table`(y)], then `ontable`(x, y)

If [`Block`(y)], then `onblock`(x, y)

For all z, if [`on`(x, z)], then $\neg$`on`(x, z)



Problem II

`ontable`/`onblock` facts are not removed automatically when deleting `on` facts

Shortcomings of eKABs (III)

Action effects ignore the ontology

$\mathcal{T} = \{\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^{-} \sqsubseteq \texttt{Block},\ \ldots\ \}$

Action $\texttt{move}(x, y)$:

Precondition:

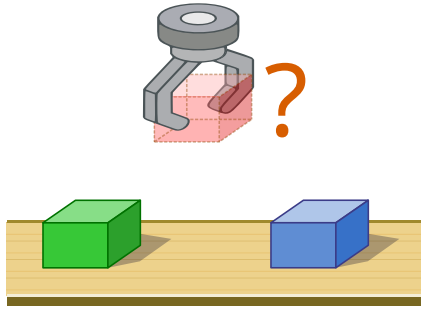
$\neg[\texttt{Blocked}(x)] \wedge \neg[\texttt{Blocked}(y)] \wedge [\texttt{Block}(x)]$

Effects:

If $[\texttt{Table}(y)]$, then $\texttt{ontable}(x, y)$

If $[\texttt{Block}(y)]$, then $\texttt{onblock}(x, y)$

For all z, if $[\texttt{on}(x, z)]$, then $\neg\texttt{on}(x, z)$



Shortcomings of eKABs (III)

Action effects ignore the ontology

$\mathcal{T} = \{\ \texttt{Block} \equiv \exists \texttt{on},\ \exists \texttt{onblock}^{-} \sqsubseteq \texttt{Block},\ \ldots\ \}$

Action $\texttt{move}(x, y)$:

Precondition:

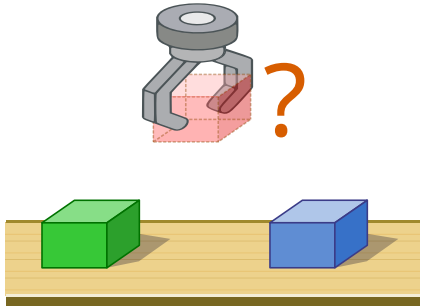
$\neg[\texttt{Blocked}(x)] \wedge \neg[\texttt{Blocked}(y)] \wedge [\texttt{Block}(x)]$

Effects:

If $[\texttt{Table}(y)]$, then $\texttt{ontable}(x, y)$

If $[\texttt{Block}(y)]$, then $\texttt{onblock}(x, y)$

For all z, if $[\texttt{on}(x, z)]$, then $\neg\texttt{on}(x, z)$



Problem III

Implicit `Block` facts can be lost if `ontable`/`onblock` facts are removed

Coherence update semantics

Given: ABox s, ground sets *Add*, *Del* (union of all applicable effects)

Coherence update semantics

Given: ABox s, ground sets *Add*, *Del* (union of all applicable effects)

Goal: ABox s' such that

- $s', \mathcal{T} \models f$ for all $f \in \mathit{Add}$
- $s', \mathcal{T} \not\models f$ for all $\neg f \in \mathit{Del}$
- $s', \mathcal{T}$ are consistent
- $s', \mathcal{T}$ preserves a maximal subset of facts entailed by $s, \mathcal{T}$ (minimal change)

Coherence update semantics

Given: ABox s, ground sets *Add*, *Del* (union of all applicable effects)

Goal: ABox s' such that

- $s', \mathcal{T} \models f$ for all $f \in \mathit{Add}$
- $s', \mathcal{T} \not\models f$ for all $\neg f \in \mathit{Del}$
- $s', \mathcal{T}$ are consistent
- $s', \mathcal{T}$ preserves a maximal subset of facts entailed by $s, \mathcal{T}$ (minimal change)

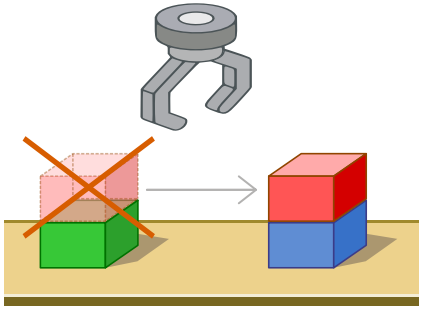
Coherence update s' for DL-Lite ontologies (Giacomo, Oriol, Rosati, and Savo 2021)

- Unique updated ABox s', unless *Add* and *Del* are contradictory
- Can be computed by stratified Datalog$^\neg$ rules $\mathcal{R}_{\mathit{Add},\mathit{Del}}$, for any ABox s

Coherent movement of blocks

$Del = \{\texttt{on}(\texttt{r}, \texttt{g})\}$

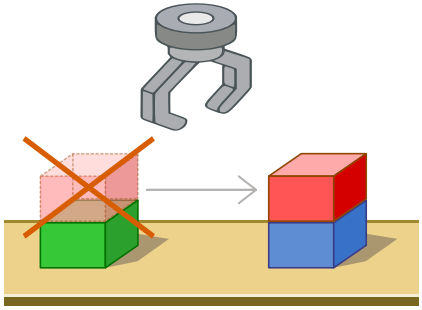
$\rightsquigarrow$ remove $\texttt{onblock}(\texttt{r}, \texttt{g})$ (II), add $\texttt{Block}(\texttt{r})$ (III)



Coherent movement of blocks

Del = {on(r, g)}
  ⇝ remove onblock(r, g) (II), add Block(r) (III)

Add = {onblock(r, b)}
  ⇝ remove onblock(r, g) (I)

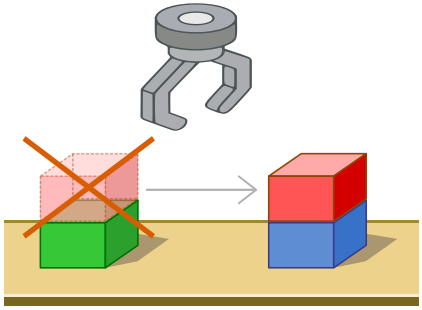


Coherent movement of blocks

Del = {on(r, g)}
  ⇝ remove onblock(r, g) (II), add Block(r) (III)

Add = {onblock(r, b)}
  ⇝ remove onblock(r, g) (I)

Action move(x, y):
  Precondition:
    ¬[Blocked(x)] ∧ ¬[Blocked(y)] ∧ [Block(x)]
  Effects:
    If [Table(y)], then ontable(x, y)
    If [Block(y)], then onblock(x, y)

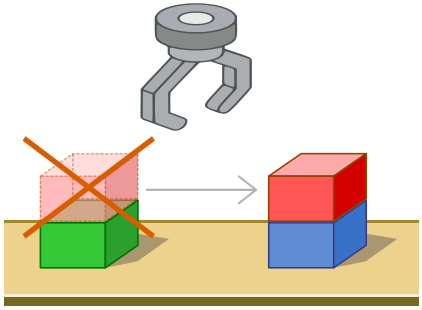


Coherent movement of blocks

Del = {on(r, g)}
  ⇝ remove onblock(r, g) (II), add Block(r) (III)

Add = {onblock(r, b)}
  ⇝ remove onblock(r, g) (I)

Action move(x, y):
  Precondition:
    ¬[Blocked(x)] ∧ ¬[Blocked(y)] ∧ [Block(x)]
  Effects:
    If [Table(y)], then ontable(x, y)
    If [Block(y)], then onblock(x, y)



- Easier to model planning problems with fully ontology-aware semantics

Coherent eKABs

Coherent eKAB (ceKAB) task: $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$

Coherent eKABs

Coherent eKAB (ceKAB) task: $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$

- Ground action a is applicable if $s, \mathcal{T} \models$ *pre* and *Add* and *Del* are not contradictory

Coherent eKABs

Coherent eKAB (ceKAB) task: $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$

- Ground action a is applicable if $s, \mathcal{T} \models \mathit{pre}$ and Add and Del are not contradictory
- Application yields the coherence update of $s, \mathit{Add}, \mathit{Del}$

Coherent eKABs

Coherent eKAB (ceKAB) task: $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$

- Ground action a is applicable if $s, \mathcal{T} \models \mathit{pre}$ and Add and Del are not contradictory
- Application yields the coherence update of $s, \mathit{Add}, \mathit{Del}$
- Compile into PDDL using derived predicates based on $\mathcal{R}_{\mathit{Add},\mathit{Del}}$

Coherent eKABs

Coherent eKAB (ceKAB) task: $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$

- Ground action a is applicable if $s, \mathcal{T} \models \mathit{pre}$ and Add and Del are not contradictory
- Application yields the coherence update of $s, \mathit{Add}, \mathit{Del}$
- Compile into PDDL using derived predicates based on $\mathcal{R}_{\mathit{Add},\mathit{Del}}$

Theorem (B, Nhu, and Röger 2025)

There is a polynomial compilation scheme from DL-Lite ceKABs into PDDL with derived predicates that preserves plan length with a factor of 2.

Coherent eKABs

Coherent eKAB (ceKAB) task: $(\mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{O}, I, G)$

- Ground action a is applicable if $s, \mathcal{T} \models \mathit{pre}$ and Add and Del are not contradictory
- Application yields the coherence update of $s, \mathit{Add}, \mathit{Del}$
- Compile into PDDL using derived predicates based on $\mathcal{R}_{\mathit{Add},\mathit{Del}}$

Theorem (B, Nhu, and Röger 2025)

There is a polynomial compilation scheme from DL-Lite ceKABs into PDDL with derived predicates that preserves plan length with a factor of 2.

- Plan existence for ceKABs is ExpSpace-complete

KR'25 experiments

- implementation of compilation using $\mathcal{R}_{Add,Del}$
- use Nemo to saturate $\mathcal{T}$ (Ivliev, Gerlach, Meusel, Steinberg, and Krötzsch 2024)

KR'25 experiments

- implementation of compilation using $\mathcal{R}_{Add,Del}$
- use Nemo to saturate $\mathcal{T}$ (Ivliev, Gerlach, Meusel, Steinberg, and Krötzsch 2024)
- benchmarks:
  - 125 previous DL-Lite eKAB tasks, slightly modified for new semantics
  - 35 new DL-Lite tasks (Blocks)

KR'25 experiments

- implementation of compilation using $\mathcal{R}_{Add,Del}$
- use Nemo to saturate $\mathcal{T}$ (Ivliev, Gerlach, Meusel, Steinberg, and Krötzsch 2024)
- benchmarks:
  - 125 previous DL-Lite eKAB tasks, slightly modified for new semantics
  - 35 new DL-Lite tasks (Blocks)
- compare to eKABs (B, Hoffmann, Kovtunova, Krötzsch, Nebel, and Steinmetz 2022)

KR'25 results

Domain	#	# solved AAAI22	# solved KR25	planning time (s) AAAI22	planning time (s) KR25
Cats*	20	20	20	32.37	0.01
Elevator	20	20	20	51.73	0.02
Robot*	20	20	18	0.03	60.17
TaskAssign	20	20	20	0.11	0.32
TPSA	15	15	15	0.95	1.03
VTA	15	15	14	2.64	3.64
VTA-Roles	15	15	14	2.66	3.82
Blocks	35	35	35	3.00	0.98
Overall	**160**	**160**	**156**	12.04	7.97

not much worse than eKAB semantics

much more expressive

sensitive to planning heuristics

Summary

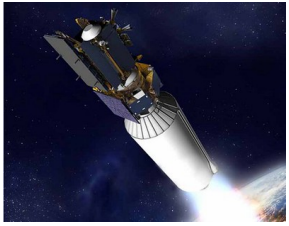
- practical compilations for planning with ontologies
- for DL-Lite and Horn DLs
- limits of compilability (Horn-$\mathcal{ALCHOIQ}$ vs. Horn-$\mathcal{SROIQ}$)

Summary

- practical compilations for planning with ontologies
- for DL-Lite and Horn DLs
- limits of compilability (Horn-$\mathcal{ALCHOIQ}$ vs. Horn-$\mathcal{SROIQ}$)

Future work:

- coherent updates beyond DL-Lite
- non-Horn languages ($\texttt{on} \sqsubseteq \texttt{onblock} \sqcup \texttt{ontable}$, $\texttt{on}$ is acyclic) (John and Koopmann 2024)
- temporal ontologies and goals



Collaborators


Jörg Hoffmann


Marcel Steinmetz


Bernhard Nebel


Alisa Kovtunova


Markus Krötzsch


Duy Nhu


Gabriele Röger

UNIVERSITÄT DES SAARLANDES

UNI FREIBURG

Dresden University of Technology

Universität Basel

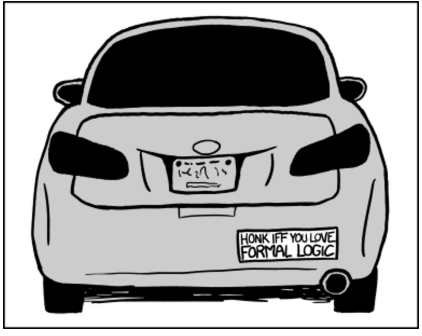
CPEC CENTER FOR PERSPICUOUS COMPUTING

DFG Deutsche Forschungsgemeinschaft German Research Foundation

Swiss National Science Foundation

Thank you!

Questions?



References I

Artale, Alessandro, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev (2009). “The DL-Lite Family and Relations”. In: *J. Artif. Intell. Res.* 36, pages 1–69. DOI: 10.1613/jair.2820.

Bienvenu, Meghyn and Magdalena Ortiz (2015). “Ontology-Mediated Query Answering with Data-Tractable Description Logics”. In: *Reasoning Web. 11th Int. Summer School*, pages 218–307. DOI: 10.1007/978-3-319-21768-0_9.

B, Stefan, Jörg Hoffmann, Alisa Kovtunova, Markus Krötzsch, Bernhard Nebel, and Marcel Steinmetz (2022). “Expressivity of Planning with Horn Description Logic Ontologies”. In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI*, pages 5503–5511. DOI: 10.1609/AAAI.V36I5.20489.

B, Stefan, Jörg Hoffmann, Alisa Kovtunova, and Marcel Steinmetz (2021). “Making DL-Lite Planning Practical”. In: *18th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR)*, pages 641–645. DOI: 10.24963/kr.2021/61.

B, Stefan, Duy Nhu, and Gabriele Röger (2025). “Automated Planning with Ontologies Under Coherence Update Semantics”. In: *22nd Int. Conf. on Principles of Knowledge Representation and Reasoning (KR)*. DOI: 10.24963/KR.2025/72.

Buhrman, Harry and Steven Homer (1992). “Superpolynomial Circuits, Almost Sparse Oracles and the Exponential Hierarchy”. In: *Proc. of the 12th Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 116–127. DOI: 10.1007/3-540-56287-7_99.

Calvanese, Diego, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati (2007). “Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family”. In: *Journal of Automated Reasoning* 39.3, pages 385–429. DOI: 10.1007/s10817-007-9078-x.

Calvanese, Diego, Marco Montali, Fabio Patrizi, and Michele Stawowy (2016). “Plan Synthesis for Knowledge and Action Bases”. In: *25th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1022–1029. URL: https://www.ijcai.org/Abstract/16/149.

References II

Carral, David, Irina Dragoste, and Markus Krötzsch (2018). "The Combined Approach to Query Answering in Horn-$\mathcal{ALCHOIQ}$". In: *16th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR)*, pages 339–348. URL: https://aaai.org/ocs/index.php/KR/KR18/paper/view/18076.

Eiter, Thomas, Carsten Lutz, Magdalena Ortiz, and Mantas Šimkus (2009). "Query Answering in Description Logics with Transitive Roles". In: *21st Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 759–764. URL: http://ijcai.org/Proceedings/09/Papers/131.pdf.

Eiter, Thomas, Magdalena Ortiz, Mantas Šimkus, Trung-Kien Tran, and Guohui Xiao (2012). "Query Rewriting for Horn-$\mathcal{SHIQ}$ Plus Rules". In: *26th AAAI Conf. on Artificial Intelligence (AAAI)*, pages 726–733. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/view/4931.

Erol, Kutluhan, Dana S. Nau, and V. S. Subrahmanian (1995). "Complexity, decidability and undecidability results for domain-independent planning". In: *Artif. Intell.* 76, pages 75–88. DOI: 10.1016/0004-3702(94)00080-K.

Giacomo, Giuseppe De, Xavier Oriol, Riccardo Rosati, and Domenico Fabio Savo (2021). "Instance-Level Update in DL-Lite Ontologies through First-Order Rewriting". In: *J. Artif. Intell. Res.* 70, pages 1335–1371. DOI: 10.1613/JAIR.1.12414.

Helmert, Malte (2006). "The Fast Downward Planning System". In: *J. Artif. Intell. Res.* 26, pages 191–246. DOI: 10.1613/jair.1705.

Hoffmann, Jörg, Ingo Weber, James Scicluna, Tomasz Kacmarek, and Anupriya Ankolekar (2008). "Combining Scalability and Expressivity in the Automatic Composition of Semantic Web Services". In: *8th Int. Conference on Web Engineering (ICWE)*. DOI: 10.1109/ICWE.2008.8.

Ivliev, Alex, Lukas Gerlach, Simon Meusel, Jakob Steinberg, and Markus Krötzsch (2024). "Nemo: Your Friendly and Versatile Rule Reasoning Toolkit". In: *21st Int. Conf. on Principles of Knowledge Representation and Reasoning (KR)*. DOI: 10.24963/KR.2024/70.

References III

John, Tobias and Patrick Koopmann (2024). “Planning with OWL-DL Ontologies”. In: *ECAI 2024 - 27th European Conference on Artificial Intelligence*, pages 4165–4172. DOI: 10.3233/FAIA240988.

Lutz, Carsten (2008). “The Complexity of Conjunctive Query Answering in Expressive Description Logics”. In: *4th Int. Joint Conf. on Automated Reasoning (IJCAR)*, pages 179–193. DOI: 10.1007/978-3-540-71070-7_16.

Nebel, Bernhard (2000). “On the Compilability and Expressive Power of Propositional Planning Formalisms”. In: *J. Artif. Intell. Res.* 12, pages 271–315. DOI: 10.1613/jair.735.

Ortiz, Magdalena, Sebastian Rudolph, and Mantas Šimkus (2010). “Worst-case Optimal Reasoning for the Horn-DL Fragments of OWL 1 and 2”. In: *12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR)*, pages 269–279. URL: https://aaai.org/ocs/index.php/KR/KR2010/paper/view/1296.

Röger, Gabriele and Claudia Grundke (2024). “Negated Occurrences of Predicates in PDDL Axiom Bodies”. In: *PuK Workshop*. URL: http://www.puk-workshop.de/rc_images/paper.pdf.

Thiébaux, Sylvie, Jörg Hoffmann, and Bernhard Nebel (2005). “In Defense of PDDL Axioms”. In: *Artif. Intell.* 168, pages 38–69. DOI: 10.1016/j.artint.2005.05.004.