Operator Pruning using Lifted Mutex Groups via Compilation on Lifted Level

Daniel Fišer

Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
danfis@danfis.cz

Representations of Classical Planning Tasks

PDDL – lifted representation

- Types: `(:types vehicle location)`
- Objects: `(:objects car1 car2 - vehicle  A B C - location)`
- Predicates: `(at ?x - vehicle ?y - location)`
- Action: `(move ?v - vehicle ?f - location ?t - location)`
- Initial state: ψ_I, Goal ψ_G

STRIPS – ground representation

- Facts: $\mathcal{F} = \{$`(at car1 A)`, `(at car1 B)`, ...$\}$
- State: $s = \{$`(at car1 A)`, ...$\} \subseteq \mathcal{F}$
- Operators $\mathcal{O}$, $o = \langle \mathrm{pre}(o) \subseteq \mathcal{F}, \mathrm{add}(o) \subseteq \mathcal{F}, \mathrm{del}(o) \subseteq \mathcal{F} \rangle$
- Initial state $I \subseteq \mathcal{F}$, Goal $G \subseteq \mathcal{F}$.

Mutex Groups in STRIPS

Mutex Group (in STRIPS)

A set of facts $M \subseteq \mathcal{F}$ is a **mutex group** if $|M \cap s| \leq 1$ for every reachable state s.

Fact-Alternating Mutex Group (in STRIPS)

A set of facts $M \subseteq \mathcal{F}$ is a **fam-group** if $|M \cap I| \leq 1$ and $|M \cap \mathrm{add}(o)| \leq |M \cap \mathrm{pre}(o) \cap \mathrm{del}(o)|$ for every operator $o \in \mathcal{O}$.

Barman: Example fam-groups

- $\mathrm{handempty}(\mathtt{hand}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shaker}_1)$
- $\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1), \mathrm{clean}(\mathtt{shot}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{cocktail}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{ingredient}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{ingredient}_2)$

Pruning Unreachable Operators in STRIPS

Facts from a mutex group are pairwise mutex, i.e., they cannot appear together in any state.

Barman: Example Pruning of Unreachable Operators

Mutex group:

- $\mathrm{handempty}(\mathtt{hand}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shaker}_1)$

Operator $\text{fill-shot}(\mathtt{shot}_1, \mathtt{i}, \mathtt{hand}_1, \mathtt{hand}_1, \mathtt{d})$ o :
$\mathrm{pre}(o) = \{\mathrm{handempty}(\mathtt{hand}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \ldots\}$

Pruning Unreachable Operators in STRIPS

Facts from a mutex group are pairwise mutex, i.e., they cannot appear together in any state.

Barman: Example Pruning of Unreachable Operators

Mutex group:

- $\text{handempty}(\texttt{hand}_1), \text{holding}(\texttt{hand}_1, \texttt{shot}_1), \text{holding}(\texttt{hand}_1, \texttt{shaker}_1)$

Operator $\text{fill-shot}(\texttt{shot}_1, \texttt{i}, \texttt{hand}_1, \texttt{hand}_1, \texttt{d})$ o :
$\text{pre}(o) = \{\text{handempty}(\texttt{hand}_1), \text{holding}(\texttt{hand}_1, \texttt{shot}_1), \ldots\}$

Pruning Dead-End Operators in STRIPS

Fam-groups can detect dead-end operators: Let o be an operator, let M be a fam-group. If $M \cap G \neq \emptyset$ and $M \cap \mathrm{pre}(o) \cap \mathrm{del}(o) \neq \emptyset$ and $M \cap \mathrm{add}(o) = \emptyset$, then o is a dead-end operator. (F & Komenda, 2018)

Barman: Example Pruning of Dead-End Operators

Fam-group M:
$\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1), \mathrm{clean}(\mathtt{shot}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{cocktail}_1), \ldots$

Goal $G = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$.

Operator $\text{empty-shot}(\mathtt{hand}_1, \mathtt{shot}_1, \mathtt{cocktail}_1)\ \ o:$
$\mathrm{pre}(o) = \{\mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{del}(o) = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{add}(o) = \{\mathrm{empty}(\mathtt{shot}_1)\}$

Pruning Dead-End Operators in STRIPS

Fam-groups can detect dead-end operators: Let o be an operator, let M be a fam-group. If $M \cap G \neq \emptyset$ and $M \cap \mathrm{pre}(o) \cap \mathrm{del}(o) \neq \emptyset$ and $M \cap \mathrm{add}(o) = \emptyset$, then o is a dead-end operator. (F & Komenda, 2018)

Barman: Example Pruning of Dead-End Operators

Fam-group M:
$\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1), \mathrm{clean}(\mathtt{shot}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{cocktail}_1), \ldots$

Goal $G = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$.

Operator $\mathrm{empty\text{-}shot}(\mathtt{hand}_1, \mathtt{shot}_1, \mathtt{cocktail}_1)$ o:
$\mathrm{pre}(o) = \{\mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{del}(o) = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{add}(o) = \{\mathrm{empty}(\mathtt{shot}_1)\}$

Pruning Dead-End Operators in STRIPS

Fam-groups can detect dead-end operators: Let o be an operator, let M be a fam-group. If $M \cap G \neq \emptyset$ and $M \cap \mathrm{pre}(o) \cap \mathrm{del}(o) \neq \emptyset$ and $M \cap \mathrm{add}(o) = \emptyset$, then o is a dead-end operator. (F & Komenda, 2018)

Barman: Example Pruning of Dead-End Operators

Fam-group M:
$\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1), \mathrm{clean}(\mathtt{shot}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{cocktail}_1), \dots$

Goal $G = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$.

Operator $\mathrm{empty\text{-}shot}(\mathtt{hand}_1, \mathtt{shot}_1, \mathtt{cocktail}_1)$ o:
$\mathrm{pre}(o) = \{\mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{del}(o) = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{add}(o) = \{\mathrm{empty}(\mathtt{shot}_1)\}$

Pruning Dead-End Operators in STRIPS

Fam-groups can detect dead-end operators: Let o be an operator, let M be a fam-group. If $M \cap G \neq \emptyset$ and $M \cap \mathrm{pre}(o) \cap \mathrm{del}(o) \neq \emptyset$ and $M \cap \mathrm{add}(o) = \emptyset$, then o is a dead-end operator. (F & Komenda, 2018)

Barman: Example Pruning of Dead-End Operators

Fam-group M:
$\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1), \mathrm{clean}(\mathtt{shot}_1), \mathrm{used}(\mathtt{shot}_1, \mathtt{cocktail}_1), \ldots$

Goal $G = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$.

Operator $\mathrm{empty\text{-}shot}(\mathtt{hand}_1, \mathtt{shot}_1, \mathtt{cocktail}_1)$ o:
$\mathrm{pre}(o) = \{\mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{del}(o) = \{\mathrm{contains}(\mathtt{shot}_1, \mathtt{cocktail}_1)\}$
$\mathrm{add}(o) = \{\mathrm{empty}(\mathtt{shot}_1)\}$

Lifted (Fact-Alternating) Mutex Groups

Lifted fam-group:

- $\mathrm{handempty}(v : \mathrm{hand}), \mathrm{holding}(v : \mathrm{hand}, c : \mathrm{conatiner})$
  where v is **fixed** variable, c is **counted** variable.

Corresponding ground fam-groups:

- $\mathrm{handempty}(\mathtt{hand}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shaker}_1)$
- $\mathrm{handempty}(\mathtt{hand}_2), \mathrm{holding}(\mathtt{hand}_2, \mathtt{shot}_1), \mathrm{holding}(\mathtt{hand}_2, \mathtt{shaker}_1)$

Lifted (Fact-Alternating) Mutex Groups

Lifted fam-group:

- $\text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$
  where v is **fixed** variable, c is **counted** variable.

Corresponding ground fam-groups:

- $\text{handempty}(\texttt{hand}_1), \text{holding}(\texttt{hand}_1, \texttt{shot}_1), \text{holding}(\texttt{hand}_1, \texttt{shaker}_1)$
- $\text{handempty}(\texttt{hand}_2), \text{holding}(\texttt{hand}_2, \texttt{shot}_1), \text{holding}(\texttt{hand}_2, \texttt{shaker}_1)$

Lifted (Fact-Alternating) Mutex Groups

Lifted fam-group:

- $\mathrm{handempty}(v : \mathrm{hand}), \mathrm{holding}(v : \mathrm{hand}, c : \mathrm{conatiner})$
  where v is **fixed** variable, c is **counted** variable.

Corresponding ground fam-groups:

- $\mathrm{handempty}(\mathtt{hand}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shot}_1), \mathrm{holding}(\mathtt{hand}_1, \mathtt{shaker}_1)$
- $\mathrm{handempty}(\mathtt{hand}_2), \mathrm{holding}(\mathtt{hand}_2, \mathtt{shot}_1), \mathrm{holding}(\mathtt{hand}_2, \mathtt{shaker}_1)$

Lifted (Fact-Alternating) Mutex Groups

Lifted fam-group:

- $\text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$
  where v is **fixed** variable, c is **counted** variable.

Corresponding ground fam-groups:

- $\text{handempty}(\texttt{hand}_1), \text{holding}(\texttt{hand}_1, \texttt{shot}_1), \text{holding}(\texttt{hand}_1, \texttt{shaker}_1)$
- $\text{handempty}(\texttt{hand}_2), \text{holding}(\texttt{hand}_2, \texttt{shot}_1), \text{holding}(\texttt{hand}_2, \texttt{shaker}_1)$

Pruning with Lifted Fam-Groups

Unifier

A **substitution** σ is a function mapping variables and objects to variables and objects so that (i) it acts as identity on objects, and (ii) mapping from variables must respect types.
Given a set of atoms A, a substitution σ is call a **unifier for** A if $\sigma(A)$ is a singleton.

Pruning with Lifted Fam-Groups

Unifier

A **substitution** σ is a function mapping variables and objects to variables and objects so that (i) it acts as identity on objects, and (ii) mapping from variables must respect types.
Given a set of atoms A, a substitution σ is call a **unifier for** A if $\sigma(A)$ is a singleton.

Pruning of Unreachable Operator

- fam-group: $\mathrm{handempty}(v : \mathrm{hand}), \mathrm{holding}(v : \mathrm{hand}, c : \mathrm{conatiner})$
- Action: $\text{fill-shot}(s : \mathrm{shot}, i : \mathrm{ingredient}, \mathtt{hand}_1, \mathtt{hand}_1, d : \mathrm{dispenser})$
  $\mathrm{pre}(a) = \{\mathrm{holding}(\mathtt{hand}_1, s), \mathrm{handempty}(\mathtt{hand}_1), \ldots\}$
- There is a unifier σ: $\sigma(v) = \mathtt{hand}_1, \sigma(c) = s$ for both $\{\mathrm{holding}(v, c), \mathrm{holding}(\mathtt{hand}_1, s)\}$ and $\{\mathrm{handempty}(v), \mathrm{handempty}(\mathtt{hand}_1)\}$.

Pruning with Lifted Fam-Groups

Unifier

A **substitution** σ is a function mapping variables and objects to variables and objects so that (i) it acts as identity on objects, and (ii) mapping from variables must respect types.
Given a set of atoms A, a substitution σ is call a **unifier for** A if $\sigma(A)$ is a singleton.

Pruning of Unreachable Operator

- fam-group: $\text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$
- Action: $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, \mathtt{hand}_1, \mathtt{hand}_1, d : \text{dispenser})$
  $\text{pre}(a) = \{\text{holding}(\mathtt{hand}_1, s), \text{handempty}(\mathtt{hand}_1), \ldots\}$
- There is a unifier σ: $\sigma(v) = \mathtt{hand}_1, \sigma(c) = s$ for both $\{\text{holding}(v, c), \text{holding}(\mathtt{hand}_1, s)\}$ and $\{\text{handempty}(v), \text{handempty}(\mathtt{hand}_1)\}$.

Pruning with Lifted Fam-Groups

Unifier

A **substitution** σ is a function mapping variables and objects to variables and objects so that (i) it acts as identity on objects, and (ii) mapping from variables must respect types.
Given a set of atoms A, a substitution σ is call a **unifier for** A if $\sigma(A)$ is a singleton.

Pruning of Unreachable Operator

- fam-group: $\mathrm{handempty}(v : \mathrm{hand}), \mathrm{holding}(v : \mathrm{hand}, c : \mathrm{conatiner})$
- Action: $\text{fill-shot}(s : \mathrm{shot}, i : \mathrm{ingredient}, \mathtt{hand}_1, \mathtt{hand}_1, d : \mathrm{dispenser})$
  $\mathrm{pre}(a) = \{\mathrm{holding}(\mathtt{hand}_1, s), \mathrm{handempty}(\mathtt{hand}_1), \ldots\}$
- There is a unifier σ: $\sigma(v) = \mathtt{hand}_1, \sigma(c) = s$ for both $\{\mathrm{holding}(v, c), \mathrm{holding}(\mathtt{hand}_1, s)\}$ and $\{\mathrm{handempty}(v), \mathrm{handempty}(\mathtt{hand}_1)\}$.

Pruning with Lifted Fam-Groups

Unifier

A **substitution** σ is a function mapping variables and objects to variables and objects so that (i) it acts as identity on objects, and (ii) mapping from variables must respect types.
Given a set of atoms A, a substitution σ is call a **unifier for** A if $\sigma(A)$ is a singleton.

Pruning of Unreachable Operator

- fam-group: $\text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$
- Action: $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, \mathtt{hand}_1, \mathtt{hand}_1, d : \text{dispenser})$
  $\text{pre}(a) = \{\text{holding}(\mathtt{hand}_1, s), \text{handempty}(\mathtt{hand}_1), \ldots\}$
- There is a unifier σ: $\sigma(v) = \mathtt{hand}_1, \sigma(c) = s$ for both $\{\text{holding}(v, c), \text{holding}(\mathtt{hand}_1, s)\}$ and $\{\text{handempty}(v), \text{handempty}(\mathtt{hand}_1)\}$.

Pruning with Lifted Fam-Groups

Unifier

A **substitution** σ is a function mapping variables and objects to variables and objects so that (i) it acts as identity on objects, and (ii) mapping from variables must respect types.
Given a set of atoms A, a substitution σ is call a **unifier for** A if $\sigma(A)$ is a singleton.

Pruning of Unreachable Operator

- fam-group: $\text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$
- Action: $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, \texttt{hand}_1, \texttt{hand}_1, d : \text{dispenser})$
  $\text{pre}(a) = \{\text{holding}(\texttt{hand}_1, s), \text{handempty}(\texttt{hand}_1), \ldots\}$
- There is a unifier σ: $\sigma(v) = \texttt{hand}_1, \sigma(c) = s$ for both $\{\text{holding}(v, c), \text{holding}(\texttt{hand}_1, s)\}$ and $\{\text{handempty}(v), \text{handempty}(\texttt{hand}_1)\}$.

Experimental Evaluation: Percentage of Pruned Operators

Percentage of Pruned Operators (IPC 2006–2018) (AAAI'20)

domain	#ps	unreach	+dead-end
agricola	20	0.00	0.00
barman*	74	15.42	**45.95**
citycar*	40	0.00	**1.61**
flashfill	18	0.10	0.10
floortile*	70	0.00	**22.79**
organic-synthesis*	7	11.44	**23.11**
parcprinter*	30	0.00	**40.83**
parking	80	3.09	3.09
scanalyzer	30	1.34	1.34
spider	15	3.47	3.47
trucks*	30	0.00	**82.14**
woodworking*	30	0.00	**10.08**
overall from above	444	3.52	**21.52**
overall	1247	1.25	**7.66**

Experimental Evaluation: Percentage of Pruned Operators

Percentage of Pruned Operators (IPC 2006–2018) (AAAI'20)

domain	#ps	unreach	+dead-end
agricola	20	0.00	0.00
barman*	74	15.42	**45.95**
citycar*	40	0.00	**1.61**
flashfill	18	0.10	0.10
floortile*	70	0.00	**22.79**
organic-synthesis*	7	11.44	**23.11**
parcprinter*	30	0.00	**40.83**
parking	80	3.09	3.09
scanalyzer	30	1.34	1.34
spider	15	3.47	3.47
trucks*	30	0.00	**82.14**
woodworking*	30	0.00	**10.08**
overall from above	444	3.52	**21.52**
overall	1247	1.25	**7.66**

We need to modify the grounding algorithm.

Unifier as a Formula

Given a substitution σ and a set of variables $V \subset \mathcal{V}$, we define:

$$\Phi^{\text{var}}_{\sigma,V} = \bigwedge_{\{v,w\} \in X} (v = w), \tag{1}$$

where $X = \{\{v, w\} \subseteq V \mid v \neq w, \sigma v = \sigma w\}$;

$$\Phi^{\text{var-obj}}_{\sigma,V} = \bigwedge_{v \in Y} (v = \sigma v), \tag{2}$$

where $Y = \{v \in V \mid \sigma v \in \mathcal{B}\}$;

$$\Phi^{\text{subtype}}_{\sigma,V} = \bigwedge_{v \in Z} \Big(\bigvee_{o \in \mathcal{D}(\tau_{var}(\sigma v))} (v = o) \Big), \tag{3}$$

where $Z = \{v \in V \mid \sigma v \notin \mathcal{B}, \tau_{var}(\sigma v) \neq \tau_{var}(v)\}$; and

$$\Phi^{\text{unifier}}_{\sigma,V} = \Phi^{\text{var}}_{\sigma,V} \land \Phi^{\text{var-obj}}_{\sigma,V} \land \Phi^{\text{subtype}}_{\sigma,V}. \tag{4}$$

Unifier as a Formula

Given a substitution σ and a set of variables $V \subset \mathcal{V}$, we define:

$$\Phi_{\sigma,V}^{\mathrm{var}} = \bigwedge_{\{v,w\} \in X} (v = w), \quad (1)$$

where $X = \{\{v, w\} \subseteq V \mid v \neq w, \sigma v = \sigma w\}$;

$$\Phi_{\sigma,V}^{\mathrm{var\text{-}obj}} = \bigwedge_{v \in Y} (v = \sigma v), \quad (2)$$

where $Y = \{v \in V \mid \sigma v \in \mathcal{B}\}$;

$$\Phi_{\sigma,V}^{\mathrm{subtype}} = \bigwedge_{v \in Z} (\bigvee_{o \in \mathcal{D}(\tau_{var}(\sigma v))} (v = o)), \quad (3)$$

where $Z = \{v \in V \mid \sigma v \notin \mathcal{B}, \tau_{var}(\sigma v) \neq \tau_{var}(v)\}$; and

$$\Phi_{\sigma,V}^{\mathrm{unifier}} = \Phi_{\sigma,V}^{\mathrm{var}} \wedge \Phi_{\sigma,V}^{\mathrm{var\text{-}obj}} \wedge \Phi_{\sigma,V}^{\mathrm{subtype}}. \quad (4)$$

Formula $\Phi_{\sigma,V}^{\mathrm{unifier}}$ captures unifier σ perfectly.

Pruning via Compilation

Lifted fam-group:
$M = \text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$

Action $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, h_1 : \text{hand}, h_2 : \text{hand}, d : \text{dispenser})$
$\text{pre}(o) = \{\text{handempty}(h_1), \text{holding}(h_2, s), \ldots\}$

Pruning via Compilation

Lifted fam-group:
$M = \text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$

Action $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, h_1 : \text{hand}, h_2 : \text{hand}, d : \text{dispenser})$
$\text{pre}(o) = \{\text{handempty}(h_1), \text{holding}(h_2, s), \ldots\}$

- Unifier σ_1, $\sigma_1(v) = \sigma_1(h_1) = x$, for $\{\text{handempty}(h_1), \text{handempty}(v)\}$

Pruning via Compilation

Lifted fam-group:
$M = \text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$

Action $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, h_1 : \text{hand}, h_2 : \text{hand}, d : \text{dispenser})$
$\text{pre}(o) = \{\text{handempty}(h_1), \text{holding}(h_2, s), \ldots\}$

- Unifier σ_1, $\sigma_1(v) = \sigma_1(h_1) = x$, for $\{\text{handempty}(h_1), \text{handempty}(v)\}$
- Unifier σ_2, $\sigma_2(x) = \sigma_2(h_2) = y$, $\sigma_2(c) = \sigma_2(s) = z$, for $\sigma_1\{\text{holding}(h_2, s), \text{holding}(v, c)\} = \{\text{holding}(h_2, s), \text{holding}(x, c)\}$

Pruning via Compilation

Lifted fam-group:
$M = \text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$

Action $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, h_1 : \text{hand}, h_2 : \text{hand}, d : \text{dispenser})$
$\text{pre}(o) = \{\text{handempty}(h_1), \text{holding}(h_2, s), \ldots\}$

- Unifier σ_1, $\sigma_1(v) = \sigma_1(h_1) = x$, for $\{\text{handempty}(h_1), \text{handempty}(v)\}$
- Unifier σ_2, $\sigma_2(x) = \sigma_2(h_2) = y$, $\sigma_2(c) = \sigma_2(s) = z$, for $\sigma_1\{\text{holding}(h_2, s), \text{holding}(v, c)\} = \{\text{holding}(h_2, s), \text{holding}(x, c)\}$
- Therefore $\text{fill-shot}(s, i, h_1, h_2, d)$ is recognized as unreachable with M **if and only if** $\Phi^{\text{unifier}}_{\sigma_1,\{h_1\}} \wedge \Phi^{\text{unifier}}_{\sigma_2\sigma_1,\{h_2,s\}}$ is true.

Pruning via Compilation

Lifted fam-group:
$M = \text{handempty}(v : \text{hand}), \text{holding}(v : \text{hand}, c : \text{conatiner})$

Action $\text{fill-shot}(s : \text{shot}, i : \text{ingredient}, h_1 : \text{hand}, h_2 : \text{hand}, d : \text{dispenser})$
$\text{pre}(o) = \{\text{handempty}(h_1), \text{holding}(h_2, s), \ldots\}$

- Unifier σ_1, $\sigma_1(v) = \sigma_1(h_1) = x$, for $\{\text{handempty}(h_1), \text{handempty}(v)\}$
- Unifier σ_2, $\sigma_2(x) = \sigma_2(h_2) = y$, $\sigma_2(c) = \sigma_2(s) = z$, for $\sigma_1\{\text{holding}(h_2, s), \text{holding}(v, c)\} = \{\text{holding}(h_2, s), \text{holding}(x, c)\}$
- Therefore $\text{fill-shot}(s, i, h_1, h_2, d)$ is recognized as unreachable with M if and only if $\Phi^{\text{unifier}}_{\sigma_1,\{h_1\}} \wedge \Phi^{\text{unifier}}_{\sigma_2\sigma_1,\{h_2,s\}}$ is true.
- Therefore, we can prune fill-shot by extending its preconditions with $\neg(\Phi^{\text{unifier}}_{\sigma_1,\{h_1\}} \wedge \Phi^{\text{unifier}}_{\sigma_2\sigma_1,\{h_2,s\}}) = (h_1 \neq h_2)$.

Pruning via Compilation

- The pruning power is exactly the same as before, but we do not need to modify the grounding algorithm.
- The pruning automatically carries to other methods, because it is directly encoded in the PDDL task. For example, successor generator for lifted planning.
- There is, however, a price to pay: The resulting formulas can inccur an exponential blow-up when normalized to conjunctions.

Experimental Results

- un: unreachability, de: dead-ends
- 56 (HTG + Optimal and satisficing IPC) domains, 3 464 tasks
- 26 domains and 1 485 tasks affcted by un
- 10 domains and 544 tasks affcted by de

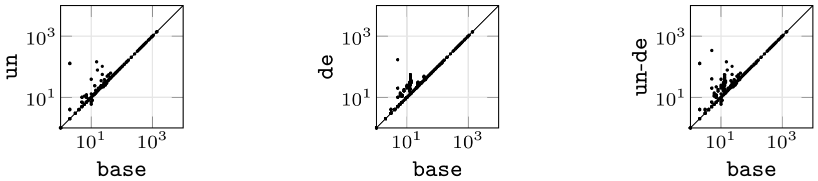

Figure: Number of normalized actions.

Experimental Results

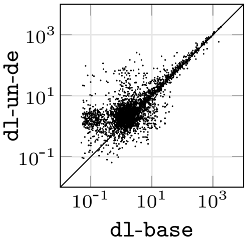

Figure: Translation time in seconds.

Experimental Results

domain	blind base	blind de	hmax base	hmax de	hadd base	hadd de	lz-hadd base	lz-hadd de
airport (50)	16	**17**	16	16	22	22	21	21
barman (74)	4	4	4	4	4	4	**6**	5
floortile (80)	2	**10**	2	2	14	14	13	**16**
parcprinter (70)	14	**18**	20	20	49	**57**	49	**57**
trucks (30)	2	**5**	3	**4**	8	**9**	9	**11**
woodworking (98)	12	**13**	0	0	0	0	0	0
others (142)	46	46	42	42	111	111	115	115
Σ (544)	96	**113**	87	**88**	208	**217**	213	**225**

Table: Number of solved tasks.

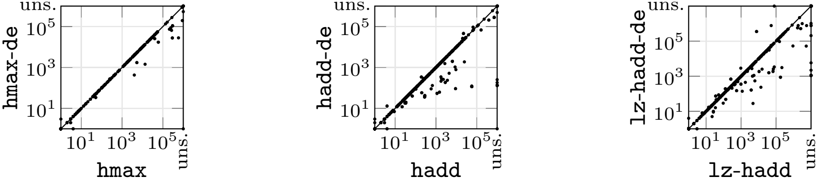

Figure: Number of dead-end states.

Experimental Results

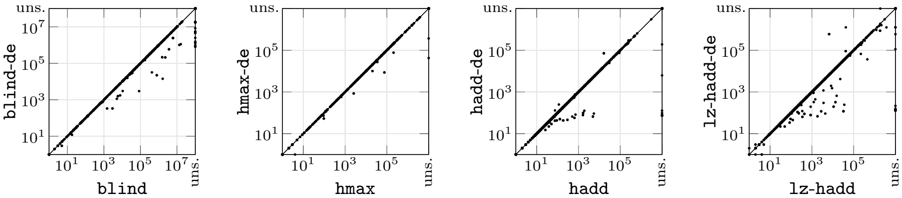

Figure: Number of expanded states (before the last f-layer for `blind` and `hmax`; all for `hadd` and `lz-hadd`).