Operator Mutexes and Symmetries for Simplifying Planning Tasks

Daniel Fišer\textsuperscript{1} \ Álvaro Torralba\textsuperscript{2} \ Alexander Shleyfman\textsuperscript{3}

\textsuperscript{1}Czech Technical University in Prague, \textit{danfis@danfis.cz}
\textsuperscript{2}Saarland University, \textit{torralba@cs.uni-saarland.de}
\textsuperscript{3}Technion – Israel Institute of Technology, \textit{alesh@campus.technion.ac.il}

January 28, 2019
A planning task in STRIPS is a tuple $\Pi = \langle F, O, I, G \rangle$ where:
- $F$ is a finite set of facts;
- $I \subseteq F$ is an initial state; $G \subseteq F$ is a goal specification;
- $O$ is a set of operators $o$ specified by $\text{pre}(o), \text{del}(o), \text{add}(o) \subseteq F$, and $c(o) \in \mathbb{R}_0^+$.
A set of operators \( O \) is **redundant** if remove \( O \) from the planning task preserves at least one (strongly optimal) plan.

- Given two redundant sets \( O_1 \) and \( O_2 \), \( O_1 \cup O_2 \) is not necessarily redundant.
- If \( O_1 \) is redundant in \( \Pi \) and \( O_2 \) is redundant in \( \Pi \setminus O_1 \), then \( O_1 \cup O_2 \) is redundant in \( \Pi \). Therefore, we can apply fixpoint computation.

**Plans:**
\[
\langle o_1, o_2 \rangle \\
\langle o_1, o_3 \rangle \\
\langle o_4, o_5 \rangle
\]

**Redundant sets:**
\[
\{ o_1, o_2 \} \\
\{ o_1, o_3 \} \\
\{ o_4, o_5 \}
\]
Redundancy

Redundant Set

A set of operators $O$ is **redundant** if remove $O$ from the planning task preserves at least one (strongly optimal) plan.

- Given two redundant sets $O_1$ and $O_2$, $O_1 \cup O_2$ is not necessarily redundant.
- If $O_1$ is redundant in $\Pi$ and $O_2$ is redundant in $\Pi \setminus O_1$, then $O_1 \cup O_2$ is redundant in $\Pi$. Therefore, we can apply fixpoint computation.

Plans: $\langle o_1, o_2 \rangle$
$\langle o_1, o_3 \rangle$
$\langle o_4, o_5 \rangle$

Redundant sets:
- $\{o_2\}$
- $\{o_3\}$
- $\{o_4, o_5\}$
Redundancy

**Redundant Set**

A set of operators $O$ is **redundant** if remove $O$ from the planning task preserves at least one (strongly optimal) plan.

- Given two redundant sets $O_1$ and $O_2$, $O_1 \cup O_2$ is not necessarily redundant.
- If $O_1$ is redundant in $\Pi$ and $O_2$ is redundant in $\Pi \setminus O_1$, then $O_1 \cup O_2$ is redundant in $\Pi$. Therefore, we can apply fixpoint computation.

**Plans:**

$\langle o_1, o_2 \rangle$
$\langle o_1, o_3 \rangle$
$\langle o_4, o_5 \rangle$

**Redundant sets:**

$\{o_3\}$
$\{o_4, o_5\}$

$\{o_3, o_4, o_5\}$ is redundant in $\Pi$
A strong operator mutex (op-mutex) $O$ is a nonempty set of operators s.t. $O \not\subseteq \pi$ for every strongly optimal plan $\pi$.

- Mutex: a set of facts $M$ s.t. $M \not\subseteq s$ for every reachable state $s$.
- What is a mutex with respect to reachable states that is a strong operator mutex with respect to strongly optimal plans.
- Definition with respect to strongly optimal plans is more general than with respect to all plans.
A strong operator mutex (op-mutex) $O$ is a nonempty set of operators s.t. $O \not\subseteq \pi$ for every strongly optimal plan $\pi$.

- In every op-mutex $O \subseteq \mathcal{O}$, there is an operator $o \in O$ such that $\{o\}$ is redundant.
- Given $O_1, O_2 \subseteq \mathcal{O}$, $O_1 \neq O_2$, if $\{o_1, o_2\}$ is an op-mutex for every $o_1 \in O_1$ and every $o_2 \in O_2$, then $O_1$ or $O_2$ is redundant.
- We proposed four different methods for inference of op-mutexes.

$$O_1 = \{o_1, \ldots, o_3\}$$
$$O_2 = \{o_4, \ldots, o_5\}$$
Inference of Operator Mutexes using Abstractions

Inference from Abstract State Space

Given an abstract state space \( \Theta^{\alpha}_{\Pi} \) and operators \( o_1, o_2 \in \mathcal{O} \), \( o_1 \neq o_2 \), if \( o_2 \) is not reachable after \( o_1 \) and \( o_1 \) is not reachable after \( o_2 \), then \( \{ o_1, o_2 \} \) is an op-mutex.

Projection to \( \{ \text{station, bus, train, arrived} \} \):

- **Station**: buy-ticket → station, ticket
- **Bus**: board-bus → bus, ticket
- **Train**: board-train → train, ticket
- **Arrived**: travel-train
- **Buy Ticket**: buy-ticket

- **Station**: buy-t → station
- **Bus**: b-bus → bus
- **Train**: b-train → train
- **Arrived**: t-arrived
- **Buy Ticket**: buy-t
Inference of Operator Mutexes using Abstractions

Inference from Abstract State Space

Given an abstract state space $\Theta_{\Pi}^\alpha$ and operators $o_1, o_2 \in \mathcal{O}$, $o_1 \neq o_2$, if $o_2$ is not reachable after $o_1$ and $o_1$ is not reachable after $o_2$, then $\{o_1, o_2\}$ is an op-mutex.

- Experimentally evaluated with projections to single fact-alternating mutex groups.
- Op-mutexes were found in 32 out of 83 domains from IPC 2006–2018.
- However, the method can be used with any abstraction (Pattern Databases, Merge-and-Shrink, Cartesian Abstractions).
Given the planning task $\Pi$, create a new auxiliary fact $f_o$ for each operator $o$, and add $f_o$ to $o$’s add effect. If $\{f_{o_1}, \ldots, f_{o_n}\}$ is a mutex in op-fact compilation, then $\{o_1, \ldots, o_n\}$ is an op-mutex in $\Pi$. 

![Diagram of operator mutexes and symmetries for simplifying planning tasks.](image-url)
Inference with Op-Fact Compilation

Given the planning task $\Pi$, create a new auxiliary fact $f_o$ for each operator $o$, and add $f_o$ to $o$’s add effect.

If $\{f_{o_1}, \ldots, f_{o_n}\}$ is a mutex in op-fact compilation, then $\{o_1, \ldots, o_n\}$ is an op-mutex in $\Pi$. 

- station
- bus, $f_{b-bus}$
- board-bus
- buy-ticket
- bus, ticket, $f_{b-bus}$, $f_{buy-t}$
- board-bus
- travel-bus
- arrived, $f_{b-bus}$, $f_{buy-t}$, $f_{t-bus}$
- arrived, $f_{b-train}$, $f_{buy-t}$, $f_{t-train}$
- train, $f_{b-train}$
- board-train
- buy-ticket
- train, ticket, $f_{b-train}$, $f_{buy-t}$
- travel-train
Operators-as-Facts Compilation

Inference with Op-Fact Compilation

Given the planning task \( \Pi \), create a new auxiliary fact \( f_o \) for each operator \( o \), and add \( f_o \) to \( o \)'s add effect.

If \( \{f_{o_1}, \ldots, f_{o_n}\} \) is a mutex in op-fact compilation, then \( \{o_1, \ldots, o_n\} \) is an op-mutex in \( \Pi \).

- Any method for inference of mutexes can be used, including \( h^m \) family of heuristics.
- Experimentally evaluated with \( h^2 \).
- Op-mutexes were found in 34 out of 83 domains from IPC 2006–2018.
- Two more methods for inference of op-mutexes can be found in the paper.
Definition (Symmetry)

A plan preserving symmetry of the transition system \( \Theta = \langle S, L, T, s_I, S_\ast \rangle \) is a permutation \( \sigma \) of \( S \cup L \) mapping states to states and labels to labels s. t.

- \( s \xrightarrow{o} s' \in T \) iff \( \sigma(s) \xrightarrow{\sigma(o)} \sigma(s') \in T \),
- \( c(o) = c(\sigma(o)) \),
- \( s \in S_\ast \) iff \( \sigma(s) \in S_\ast \), and
- \( \sigma(s_I) = s_I \)

for all states \( s, s' \in S \) and all labels \( o \in L \).

For every plan \( \pi \) it holds that \( \sigma(\pi) \) is also a plan, moreover \( |\pi| = |\sigma(\pi)| \) and \( c(\pi) = c(\sigma(\pi)) \).
Pruning with Operator Mutexes and Symmetries

Proving Redundancy

- If \( \{o_1, o_2\} \) is an op-mutex and there exists a symmetry \( \sigma(o_1) = o_2 \), then both \( \{o_1\} \) and \( \{o_2\} \) are redundant.

- And similarly for two sets of operators \( O_1 \) and \( O_2 \): If \( \{o_1, o_2\} \) is an op-mutex for every \( o_1 \in O_1 \) and every \( o_2 \in O_2 \), and there exists symmetry \( \sigma(O_1) = O_2 \), then both \( O_1 \) and \( O_2 \) are redundant.
Pruning with Operator Mutexes and Symmetries

Proving Redundancy

- If \( \{o_1, o_2\} \) is an op-mutex and there exists a symmetry \( \sigma(o_1) = o_2 \), then both \( \{o_1\} \) and \( \{o_2\} \) are redundant.

- And similarly for two sets of operators \( O_1 \) and \( O_2 \): If \( \{o_1, o_2\} \) is an op-mutex for every \( o_1 \in O_1 \) and every \( o_2 \in O_2 \), and there exists symmetry \( \sigma(O_1) = O_2 \), then both \( O_1 \) and \( O_2 \) are redundant.
Pruning with Operator Mutexes and Symmetries

Proving Redundancy

- If \( \{o_1, o_2\} \) is an op-mutex and there exists a symmetry \( \sigma(o_1) = o_2 \), then both \( \{o_1\} \) and \( \{o_2\} \) are redundant.

- And similarly for two sets of operators \( O_1 \) and \( O_2 \): If \( \{o_1, o_2\} \) is an op-mutex for every \( o_1 \in O_1 \) and every \( o_2 \in O_2 \), and there exists symmetry \( \sigma(O_1) = O_2 \), then both \( O_1 \) and \( O_2 \) are redundant.

- Each pruning step destroys at least one symmetry, therefore the number of steps is limited by the number of symmetries.

- Finding a complete bipartite subgraph is NP-hard.

- Therefore, we use a greedy approach for inference of redundant sets: We choose the largest set of operators that destroys the least number of symmetries in a hope that we will be able to do more steps.
## Pruning Results

<table>
<thead>
<tr>
<th>domain</th>
<th>baseline h²+de</th>
<th>fam</th>
<th>Π²₀ᵖ</th>
<th>h²+de with fam</th>
<th>Π²₀ᵖ</th>
</tr>
</thead>
<tbody>
<tr>
<td>barman11</td>
<td>52.11</td>
<td>4.71</td>
<td>24.48</td>
<td>+3.60</td>
<td>+3.60</td>
</tr>
<tr>
<td>barman14</td>
<td>53.43</td>
<td>4.94</td>
<td>25.28</td>
<td>+4.01</td>
<td>+4.01</td>
</tr>
<tr>
<td>caldera18</td>
<td>39.34</td>
<td>0.73</td>
<td>13.25</td>
<td>0.00</td>
<td>+1.45</td>
</tr>
<tr>
<td>cavediving14</td>
<td>0.66</td>
<td>1.15</td>
<td>1.44</td>
<td>+1.15</td>
<td>+1.15</td>
</tr>
<tr>
<td>childsnack14</td>
<td>0.00</td>
<td>39.58</td>
<td>39.58</td>
<td>+39.58</td>
<td>+39.58</td>
</tr>
<tr>
<td>hiking14</td>
<td>0.00</td>
<td>0.07</td>
<td>0.07</td>
<td>+0.07</td>
<td>+0.07</td>
</tr>
<tr>
<td>parcprinter08</td>
<td>70.68</td>
<td>28.61</td>
<td>28.61</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>parcprinter11</td>
<td>70.15</td>
<td>25.75</td>
<td>25.75</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>pathways06</td>
<td>3.94</td>
<td>2.22</td>
<td>2.26</td>
<td>+3.73</td>
<td>+3.73</td>
</tr>
<tr>
<td>pegsol08</td>
<td>13.57</td>
<td>0.00</td>
<td>1.92</td>
<td>+0.36</td>
<td>+0.36</td>
</tr>
<tr>
<td>pegsol11</td>
<td>9.53</td>
<td>1.08</td>
<td>3.51</td>
<td>+0.20</td>
<td>+0.20</td>
</tr>
<tr>
<td>pipesworld06</td>
<td>11.46</td>
<td>0.04</td>
<td>4.04</td>
<td>+0.04</td>
<td>+0.04</td>
</tr>
<tr>
<td>scanalyzer08</td>
<td>3.19</td>
<td>0.00</td>
<td>2.10</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>scanalyzer11</td>
<td>3.17</td>
<td>0.00</td>
<td>2.12</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>sokoban08</td>
<td>0.24</td>
<td>1.54</td>
<td>1.71</td>
<td>+1.54</td>
<td>+1.60</td>
</tr>
<tr>
<td>sokoban11</td>
<td>0.35</td>
<td>1.86</td>
<td>2.12</td>
<td>+1.86</td>
<td>+1.95</td>
</tr>
<tr>
<td>tpp06</td>
<td>37.96</td>
<td>1.49</td>
<td>19.97</td>
<td>+1.79</td>
<td>+1.79</td>
</tr>
<tr>
<td>trucks06</td>
<td>75.02</td>
<td>0.00</td>
<td>0.71</td>
<td>+7.87</td>
<td>+7.87</td>
</tr>
<tr>
<td>woodworking08</td>
<td>53.47</td>
<td>2.83</td>
<td>10.79</td>
<td>+1.60</td>
<td>+1.60</td>
</tr>
<tr>
<td>woodworking11</td>
<td>53.72</td>
<td>2.73</td>
<td>10.17</td>
<td>+1.83</td>
<td>+1.83</td>
</tr>
<tr>
<td>overall</td>
<td>53.72</td>
<td>7.99</td>
<td>13.73</td>
<td>+4.29</td>
<td>+4.39</td>
</tr>
</tbody>
</table>

**Table:** Average percentage of removed operators with 15 minutes time limit.
Conclusion

- We introduced a new concept, called operator mutex, stating that a certain set of operators cannot appear together in the same plan.
- We proposed four different methods for inference of op-mutexes.
- We showed that even the simplest variants of the inference methods were able to find op-mutexes in a sizeable amount of domains.
- We combined op-mutexes with structural symmetries to prove certain sets of operators to be redundant so they can be safely removed from the planning task.