Tunneling and Decomposition-Based State Reduction for Optimal Planning

Raz Nissim   Udi Apsel   Ronen Brafman

Ben-Gurion University of the Negev, Beer-Sheva, Israel

August XX, 2012
Contributions

In a nutshell

- Two state-space reduction techniques which preserve optimality of search.
- State-space reduction is achieved by pruning the *actions* applicable at state $s$, based on the action leading to $s$.
- Both methods are effective in reducing search time, incur very little overhead, and are easy to implement.
Main Idea – Action Tunneling

If the action leading to our current state is $a$, such that:

- $a$ has a single effect, changing the value of variable $p$ to $v_1$.
- $v_1$ isn’t $p$’s goal value.
- any action that changes $p$ from $v_1$ to some other value does not affect other variables.

*Then, after $a$, we can prune all actions except those that change $p$.***
Main Idea – Partition-Based Pruning

Given a (disjoint) partition of the set of actions, if

- $a$ only affects actions in its own partition.
- $a$ does not achieve some goal.

Then, after $a$, we can prune all actions not in $a$'s partition.
The model

**Definition (SAS+)**

A SAS+ planning problem is given as a tuple

\[ \Pi = \langle \mathcal{V}, s_0, s_\star, \mathcal{A} \rangle \]

- Each \( a = \langle \text{pre}, \text{prevail}, \text{eff} \rangle \in \mathcal{A} \) is given by its preconditions, prevail conditions (required, but not affected by the action) and effects.
Action tunneling – definitions

**Action a allows a tunnel** if the following conditions hold:

1. \( \exists (v_i, p) \in eff \text{ such that } (v_i, p') \in s_\ast \text{ and } p \neq p' \).
2. Any action having a prevail \((v_i, p_i) \in eff\), has some pre-post condition \((v_j, p_j, p'_j)\) such that \((v_j, p_j) \in eff\).
3. For any action \(a'\) having a pre-post condition \((v_i, p_i, p'_i)\) where \((v_i, p_i) \in eff\), then:
   - the preconditions of \(a'\) are satisfied by \(s_{min}(a)\).
   - \(a'\) affects only the variables affected by \(a\).
### Action tunneling – definitions

**Action a allows a tunnel** if the following conditions hold:

1. \( \exists (v_i, p) \in \text{eff} \) such that \( (v_i, p') \in s_* \) and \( p \neq p' \).
2. Any action having a prevail \( (v_i, p_i) \in \text{eff} \), has some pre-post condition \( (v_j, p_j, p'_j) \) such that \( (v_j, p_j) \in \text{eff} \).
3. For any action \( a' \) having a *pre-post* condition \( (v_i, p_i, p'_i) \) where \( (v_i, p_i) \in \text{eff} \), then:
   - the preconditions of \( a' \) are satisfied by \( s_{\min}(a) \).
   - \( a' \) affects only the variables affected by \( a \).

- If \( a \) allows a tunnel, define \( \text{tunnel}(a) \) to be all operators which require \( <v_i, p'_i> \in \text{eff} \) as a pre/prevail condition.
- Otherwise, \( \text{tunnel}(a) = \mathcal{A} \).
**Action tunneling – pruning rule**

**Definition (Action tunneling pruning rule)**

Following action $a$, prune all actions not in $tunnel(a)$.

A valid sequence of actions $\pi = (a_1, a_2 \ldots, a_k)$ is said to be **tunnel-pruned** if for some action $a_i$, $a_{i+1} \notin tunnel(a_i)$. 
Running example

Action graph

A* search space

A* + Tunneling
Running example

Action graph

\[ a_3 = \langle \{ v_2 = 0 \}, \{ \}, \{ v_2 = 1 \} \rangle \]
\[ a_4 = \langle \{ v_2 = 1 \}, \{ \}, \{ v_2 = 2 \} \rangle \]
\[ a_5 = \langle \{ v_4 = 0 \}, \{ v_1 = 2, v_2 = 1 \}, \{ v_4 = 1 \} \rangle \]
\[ a_8 = \langle \{ v_4 = 1 \}, \{ v_3 = 2 \}, \{ v_4 = 2 \} \rangle \]
\[ a_6 = \langle \{ v_1 = 0 \}, \{ \}, \{ v_3 = 1 \} \rangle \]
\[ a_7 = \langle \{ v_3 = 1 \}, \{ \}, \{ v_3 = 2 \} \rangle \]

A* search space

A* + Tunneling

Nissim, Apsel, Brafman  Tunneling and Decomposition-Based State Reduction for Optimal Planning   8/18
Running example

Action graph

- \( a_3 = \langle v_2 = 0 \rangle, \langle \rangle, \langle v_2 = 1 \rangle \)
- \( a_1 = \langle v_1 = 0 \rangle, \langle \rangle, \langle v_1 = 1 \rangle \)
- \( a_4 = \langle v_2 = 1 \rangle, \langle \rangle, \langle v_2 = 2 \rangle \)
- \( a_2 = \langle v_1 = 1 \rangle, \langle \rangle, \langle v_1 = 2 \rangle \)
- \( a_5 = \langle v_4 = 0 \rangle, \langle v_1 = 2, v_2 = 2 \rangle, \langle v_4 = 1 \rangle \)
- \( a_8 = \langle v_4 = 1 \rangle, \langle v_3 = 2 \rangle, \langle v_4 = 2 \rangle \)
- \( a_6 = \langle v_1 = 0 \rangle, \langle \rangle, \langle v_3 = 1 \rangle \)
- \( a_7 = \langle v_3 = 1 \rangle, \langle \rangle, \langle v_3 = 2 \rangle \)

A* search space

A* + Tunneling
Main contributions and intuition

- **Action tunneling**
- **Partition-based pruning**
- **Action decomposition**
- **Empirical results**

Running example

**Action graph**

- $a_3 = \{v_2 = 0\}, \{\}, \{v_2 = 1\}$
- $a_1 = \{v_1 = 0\}, \{\}, \{v_1 = 1\}$
- $a_4 = \{v_2 = 1\}, \{\}, \{v_2 = 2\}$
- $a_2 = \{v_1 = 1\}, \{\}, \{v_1 = 2\}$
- $a_5 = \{v_4 = 0\}, \{v_1 = 2, v_2 = 2\}, \{v_4 = 1\}$
- $a_8 = \{v_4 = 1\}, \{v_3 = 2\}, \{v_4 = 2\}$
- $a_6 = \{v_3 = 0\}, \{\}, \{v_3 = 1\}$
- $a_7 = \{v_3 = 1\}, \{\}, \{v_3 = 2\}$

**A* search space**

**A* + Tunneling**

Nissim, Apsel, Brafman  Tunneling and Decomposition-Based State Reduction for Optimal Planning  8/18
The model

**Definition (Partitioned SAS\(^+\))**

Standard SAS\(^+\) planning problem where the actions are partitioned into \(i\) disjoint sets (\(\mathcal{A} = \{\mathcal{A}_i\}_{i=1}^k\)).

**Public vs. private actions**

The partition induces a distinction of actions and variables as *public* (affecting/affected by more than one agent) and *private* (affecting/affected by a single agent).
The model

Definition (Partitioned SAS$^+$)

Standard SAS$^+$ planning problem where the actions are partitioned into $i$ disjoint sets ($\mathcal{A} = \{\mathcal{A}_i\}_{i=1}^k$).

Public vs. private actions

The partition induces a distinction of actions and variables as public (affecting/affected by more than one agent) and private (affecting/affected by a single agent).

Example - Logistics

- A vehicle’s *move* actions are private since they affect only the agent performing them.
- *Load/unload* actions are public as they may achieve or destroy other agents’ preconditions.
Partition-based pruning rule

Definition (Partition-based pruning rule)
Following private action $a \in A_i$, prune all actions not in $A_i$.

A valid sequence of actions $\pi = (a_1, a_2 \ldots, a_k)$ is said to be **PB-pruned** if for some private action $a_i \in A_i$, $a_{i+1} \notin A_i$. 
Running example

Partitioned action graph

\[ a_3 = \langle v_2 = 0, \emptyset, \{v_1 = 1\} \rangle \]
\[ a_1 = \langle v_1 = 0, \emptyset, \{v_1 = 1\} \rangle \]
\[ a_4 = \langle v_2 = 1, \emptyset, \{v_2 = 2\} \rangle \]
\[ a_2 = \langle v_1 = 1, \emptyset, \{v_1 = 2\} \rangle \]
\[ a_5 = \langle v_4 = 0, \{v_1 = 2, v_2 = 2\}, \{v_4 = 1\} \rangle \]
\[ a_8 = \langle v_4 = 1, \{v_3 = 2\}, \{v_4 = 2\} \rangle \]
\[ a_6 = \langle v_3 = 0, \emptyset, \{v_3 = 1\} \rangle \]
\[ a_7 = \langle v_3 = 1, \emptyset, \{v_3 = 2\} \rangle \]

A* search space

A* + PB
Running example

Partitioned action graph

A1

a3=\{v2=0,\} \{v2=1\}

a1=\{v1=0,\} \{v1=1\}

a4=\{v2=1,\} \{v2=2\}

a2=\{v1=1,\} \{v1=2\}

a5=\{v4=0,\} \{v1=2, v2=2\} \{v4=1\}

a8=\{v4=1,\} \{v3=2\} \{v4=2\}

a6=\{v3=0,\} \{v3=1\}

a7=\{v3=1,\} \{v3=2\}

A* search space

A* + PB

Nissim, Apsel, Brafman  Tunneling and Decomposition-Based State Reduction for Optimal Planning  11/18
Running example
Running example

Partitioned action graph

A1

\[
a_3 = \langle v2=0, \{v2=1\} >
\]

\[
a_1 = \langle v1=0, \{v1=1\} >
\]

\[
a_4 = \langle v2=1, \{v2=2\} >
\]

\[
a_2 = \langle v1=1, \{v1=2\} >
\]

\[
a_5 = \langle v4=0, \{v1=2,v2=2\}, \{v4=1\} >
\]

A2

\[
a_8 = \langle v4=1, \{v3=2\}, \{v4=2\} >
\]

\[
a_6 = \langle v3=0, \{v3=1\} >
\]

\[
a_7 = \langle v3=1, \{v3=2\} >
\]

A* search space

A* + PB

Nissim, Apsel, Brafman  Tunneling and Decomposition-Based State Reduction for Optimal Planning  11/18
Tunneling and PB pruning

- The two methods do not prune the same states.
- Combining them as is does not maintain optimality.
- The following pruning rule does maintain optimality:

**Definition (Tunnel+PB pruning rule)**

Always perform PB pruning, and apply tunnel pruning only if the creating action is private.
Running example

Partitioned action graph $A_1$

- $a_3 = \langle v_2 = 0, \emptyset, \emptyset, v_1 = 1 \rangle$
- $a_1 = \langle v_1 = 0, \emptyset, \emptyset, v_1 = 1 \rangle$
- $a_4 = \langle v_2 = 1, \emptyset, \emptyset, v_2 = 2 \rangle$
- $a_2 = \langle v_1 = 1, \emptyset, \emptyset, v_1 = 2 \rangle$
- $a_5 = \langle v_4 = 0, v_1 = 2, v_2 = 2, v_4 = 1 \rangle$

$A^* + Tunneling$

- States $000, 010, 011, 001$
- Actions $a_1, a_3, a_6, a_7$

$A^* + PB$

- States $000, 010, 011, 001$
- Actions $a_1, a_3, a_6, a_7$

$A^* + PB + T$

- States $000, 010, 011, 001$
- Actions $a_1, a_3, a_6, a_7$
How do we partition the actions?

- Easy, if the domain has *multi-agent* structure.
- However, most IPC domains do not exhibit such structure.

**Fundamental question:**
How do we evaluate the effectiveness (in terms of pruning) of a given partition?
Effective decomposition of the actions

- An exponential number of possible partitions exist.
- It is not known a priori how effective PB pruning will be using a certain partition.
- We need an *approximate* measure of partition quality.
Effective decomposition of the actions

- An exponential number of possible partitions exist.
- It is not known a priori how effective PB pruning will be using a certain partition.
- We need an *approximate* measure of partition quality.

**Symmetry score (Γ)**

\[
\Gamma(\{A\}_{i=1}^{k}) = \sum_{i=1}^{k} (pr(a \in A_i \text{ and } a \text{ is private}) \times pr(a \notin A_i))
\]

- Quick to compute.
- Correlates well with PB pruning effectiveness.
Correlation between $\Gamma$ and running time using PB pruning

![Plot showing correlation between $\Gamma$ and running time using PB pruning. The x-axis represents the Symmetry Score, and the y-axis represents the Runtime/Max Runtime. The plot includes data points for logistics 9-0, rovers 5, satellites 6, and zenotravel 9.](image-url)
# Experimental results

<table>
<thead>
<tr>
<th>Domain</th>
<th>Coverage</th>
<th>Total Time</th>
<th>Search Time</th>
<th>Expanded</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A*</td>
<td>PB</td>
<td>T</td>
<td>PBT</td>
</tr>
<tr>
<td>airport</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td>blocks</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>depot</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>driverlog</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>freecell</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>grid</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>gripper</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>logistics00</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>logistics98</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>miconic</td>
<td>141</td>
<td>141</td>
<td>141</td>
<td>141</td>
</tr>
<tr>
<td>mprime</td>
<td>23</td>
<td>20</td>
<td>22</td>
<td>20</td>
</tr>
<tr>
<td>mystery</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>pathways-noneg</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>pipes-notankage</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>psr-small</td>
<td>49</td>
<td>49</td>
<td>49</td>
<td>49</td>
</tr>
<tr>
<td>rovers</td>
<td>7</td>
<td>8</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>satellite</td>
<td>7</td>
<td>12</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>tpp</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>transport</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>zenotravel</td>
<td>12</td>
<td>13</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td><strong>Total/GeoMean</strong></td>
<td>441</td>
<td>445</td>
<td>440</td>
<td>445</td>
</tr>
</tbody>
</table>
Future work

1. Do combinations of these and other pruning methods maintain optimality?

2. Provide sufficient conditions so that two pruning methods could be combined.

3. Find better measures of partition effectiveness and improve the partitioning process.