Tunneling and Decomposition-Based State Reduction for Optimal Planning

Raz Nissim, Udi Apsel and Ronen Brafman

Manuel Heusner

Universität Basel

December 20, 2012

UNI BASEL

Motivation

What is pruning?

- Only apply subset of actions at a state
- Reduce the state space

Why do we need pruning?

- Heuristics for generalized planners have limited strength (Helmert&Röger,2008)
- Only a few generalized prunig methods exist
- Faster search and larger coverage

Pruning methods

- Action Tunneling
- Partition-Based Path Pruning
- PB-Pruning and Tunneling

Idea and properties of pruning methods in this presentation

Idea:

- Group related actions together
- Prevent interleaving with other actions

Properties:

- Generalize earlier pruning methods (Tunneling in Sokoban)
- Preserve optimality and completeness
- Are independent of heuristics

The model

Definition (SAS$^+$)

SAS$^+$ is a planning task formalism given as a tuple $\Pi = \langle \mathcal{V}, s_0, s_\star, \mathcal{A} \rangle$

- $\mathcal{V}$: a set of *variables* which having finite domains D_v
- s_0: *initial state*
- $s_\star$: an assignment of variables in $\mathcal{G} \subseteq \mathcal{V}$ denoting the goal conditions
- $\mathcal{A}$: set of *actions* $a = \langle pre, eff \rangle$

The idea and the pruning rule

- **If** the action a leading to the current state sets a goal variable $v \mapsto p$

The idea and the pruning rule

- **If** the action a leading to the current state sets a goal variable $v \mapsto p$
- **and** p is not the goal value of v

The idea and the pruning rule

- **If** the action a leading to the current state sets a goal variable $v \mapsto p$
- **and** p is not the goal value of v
- **and** all actions that have precondition $v \mapsto p$ do not affect other variables than action a

The idea and the pruning rule

- **If** the action a leading to the current state sets a goal variable $v \mapsto p$
- **and** p is not the goal value of v
- **and** all actions that have precondition $v \mapsto p$ do not affect other variables than action a
- **then** apply only actions that have precondition $v \mapsto p$

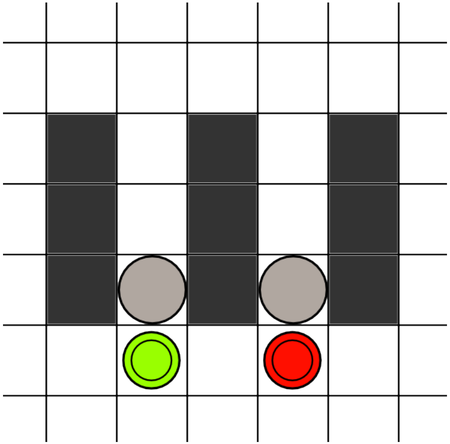
The idea and the pruning rule

- **If** the action a leading to the current state sets a goal variable $v \mapsto p$
- **and** p is not the goal value of v
- **and** all actions that have precondition $v \mapsto p$ do not affect other variables than action a
- **then** apply only actions that have precondition $v \mapsto p$

Action Tunneling Pruning Rule

Following action a, prune all actions not in $\mathit{tunnel}(a)$.

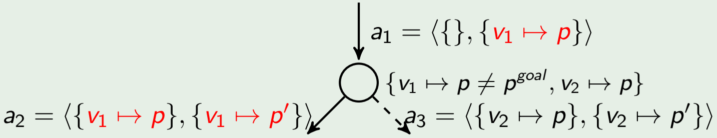
The idea and the pruning rule (example)



Pruning Example

Example

$tunnel(a_1) = \{a_2\}$ (a_1 allows a tunnel)


$a_1 = \langle\{\}, \{v_1 \mapsto p\}\rangle$
$\{v_1 \mapsto p \neq p^{goal}, v_2 \mapsto p\}$
$a_2 = \langle\{v_1 \mapsto p\}, \{v_1 \mapsto p'\}\rangle$
$a_3 = \langle\{v_2 \mapsto p\}, \{v_2 \mapsto p'\}\rangle$

Pruning Example

Example

$tunnel(a_1) = \{a_2\}$ (a_1 allows a tunnel)

$$\begin{tikzcd}
 & {} \arrow[d, "a_1 = \langle \{\}, \{v_1 \mapsto p\} \rangle"] & \\
 & \bigcirc \ \{v_1 \mapsto p \neq p^{goal}, v_2 \mapsto p\} \arrow[dl, "a_2 = \langle \{v_1 \mapsto p\}, \{v_1 \mapsto p'\} \rangle"'] \arrow[dr, dashed, "a_3 = \langle \{v_2 \mapsto p\}, \{v_2 \mapsto p'\} \rangle"] & \\
 {} & & {}
\end{tikzcd}$$

$tunnel(a_4) = \mathcal{A}$ (a_4 does not allow a tunnel)

$$\begin{tikzcd}
 & {} \arrow[d, "a_4 = \langle \{\}, \{v_3 \mapsto q\} \rangle"] & \\
 & \bigcirc \ \{v_3 \mapsto q \neq q^{goal}, v_4 \mapsto q\} \arrow[dl, "a_5 = \langle \{v_3 \mapsto q\}, \{v_3 \mapsto q'\} \rangle"'] \arrow[d, "a_6 = \langle \{v_3 \mapsto q\}, \{v_3 \mapsto q', v_4 \mapsto q'\} \rangle"] \arrow[dr, "a_7 = \langle \{v_4 \mapsto q\}, \{v_4 \mapsto q'\} \rangle"] & \\
 {} & {} & {}
\end{tikzcd}$$

The idea and the pruning rule

- Partition actions into disjoint sets
- Private actions only affect actions in own partition
- Public actions affect actions in other partitions

The idea and the pruning rule

- Partition actions into disjoint sets
- Private actions only affect actions in own partition
- Public actions affect actions in other partitions

- **If** the action leading to the current state is private
- **and** the state is not the goal state
- **then** only apply actions, which are in the same partition.

The idea and the pruning rule

- Partition actions into disjoint sets
- Private actions only affect actions in own partition
- Public actions affect actions in other partitions

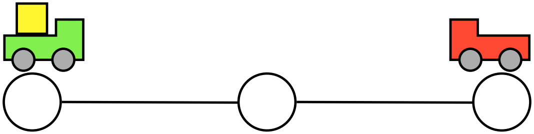
- **If** the action leading to the current state is private
- **and** the state is not the goal state
- **then** only apply actions, which are in the same partition.

Partition-Based Pruning Rule

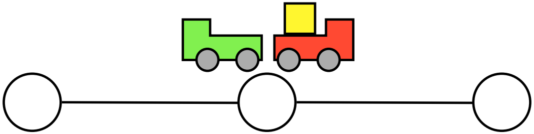
Following a private action $a \in \mathcal{A}_i$, prune all actions not in $\mathcal{A}_i$.

The idea and the pruning rule (example) (1)

Init:



Goal:



The idea and the pruning rule (example) (2)

Example

1. Partition:

private

- $drive\text{–}t1_{xy} = \langle\{truck1 \mapsto x\}, \{truck1 \mapsto y\}\rangle$

public

- $load\text{–}t1_{x} = \langle\{truck1 \mapsto x, packet \mapsto x\}, \{packet \mapsto t1\}\rangle$
- $unload\text{–}t1_{x} = \langle\{truck1 \mapsto x, packet \mapsto t1\}, \{packet \mapsto x\}\rangle$

2. Partition:

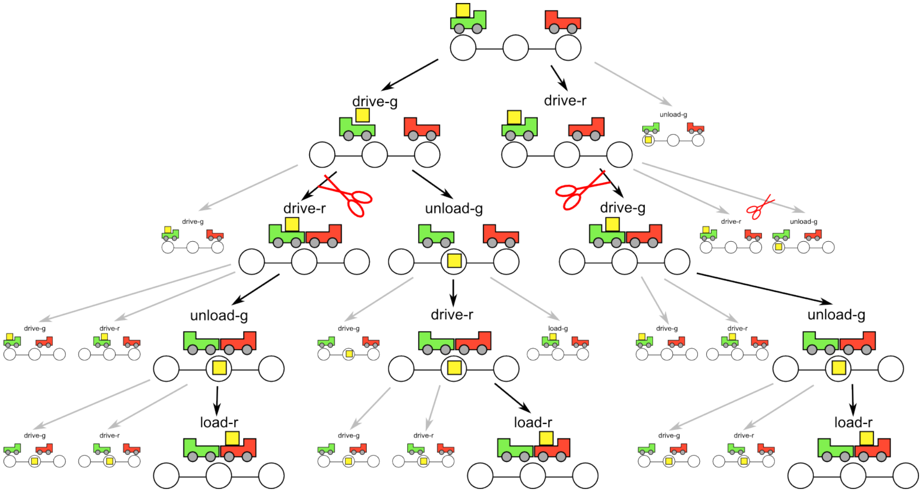
private

- $drive\text{–}t2_{xy} = \langle\{truck2 \mapsto x\}, \{truck2 \mapsto y\}\rangle$

public

- $load\text{–}t2_{x} = \langle\{truck2 \mapsto x, packet \mapsto x\}, \{packet \mapsto t2\}\rangle$
- $unload\text{–}t2_{x} = \langle\{truck2 \mapsto x, packet \mapsto t2\}, \{packet \mapsto x\}\rangle$

Pruning Example



The idea and the pruning rule

- Action Tunneling and Partition Based Path Pruning do not prune the same states
- Combine them and preserve optimality

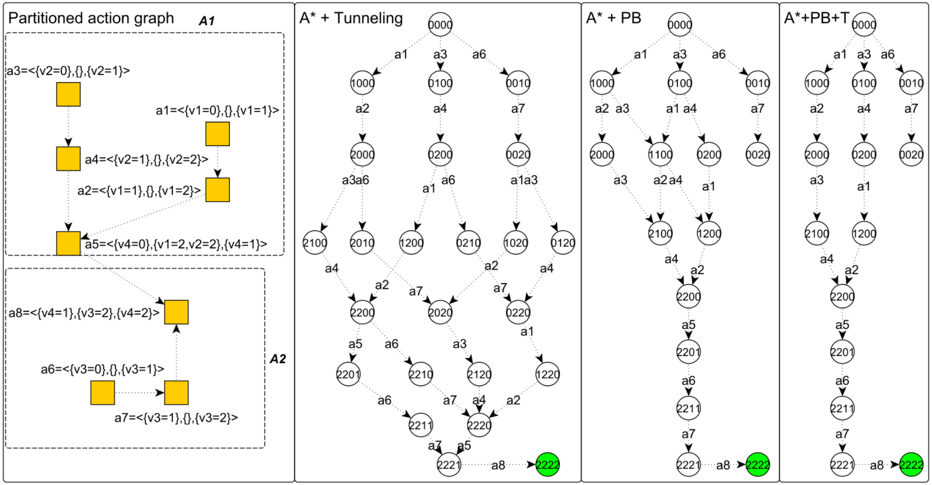
The idea and the pruning rule

- Action Tunneling and Partition Based Path Pruning do not prune the same states
- Combine them and preserve optimality

Action Tunneling and Partition-Based Pruning Rule

Always apply the Partition-Based pruning rule and apply the Tunneling pruning rule only if the creating action is private.

Pruning example



Search algorithm

- PP-A* + h^{lm-cut}
- PP-A* is an extension of A*
  - Reexpand node if another action leads to the same state with the same path cost
- Guarantees optimality while using path dependend action pruning

Search algorithm

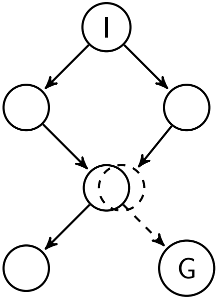
- PP-A* + $h^{lm\text{–}cut}$
- PP-A* is an extension of A*
  - Reexpand node if another action leads to the same state with the same path cost
- Guarantees optimality while using path dependend action pruning

Open question

Is A* optimal in combination with path dependend pruning methods?
$\rightsquigarrow$ The source of concern is duplicate elimination

Search algorithm

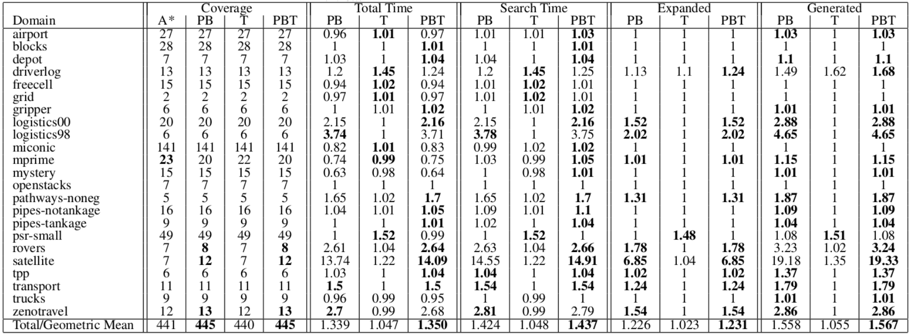
- PP-A* + $h^{lm\text{–}cut}$
- PP-A* is an extension of A*
  - Reexpand node if another action leads to the same state with the same path cost
- Guarantees optimality while using path dependend action pruning



Open question

Is A* optimal in combination with path dependend pruning methods?
$\rightsquigarrow$ The source of concern is duplicate elimination

Results



Domain	Coverage				Total Time			Search Time			Expanded			Generated		
	A*	PB	T	PBT	PB	T	PBT	PB	T	PBT	PB	T	PBT	PB	T	PBT
airport	27	27	27	27	0.96	**1.01**	0.97	1.01	1.01	**1.03**	1	1	1	**1.03**	1	**1.03**
blocks	28	28	28	28	1	1	**1.01**	1	1	**1.01**	1	1	1	1	1	1
depot	7	7	7	7	1.03	1	**1.04**	1.04	1	**1.04**	1	1	1	**1.1**	1	**1.1**
driverlog	13	13	13	13	1.2	**1.45**	1.24	1.2	**1.45**	1.25	1.13	1.1	**1.24**	1.49	1.62	**1.68**
freecell	15	15	15	15	0.94	**1.02**	0.94	1.01	**1.02**	1.01	1	1	1	1	1	1
grid	2	2	2	2	0.97	**1.01**	0.97	1.01	**1.02**	1.01	1	1	1	1	1	1
gripper	6	6	6	6	1	1.01	**1.02**	1	1.01	**1.02**	1	1	1	**1.01**	1	**1.01**
logistics00	20	20	20	20	2.15	1	**2.16**	2.15	1	**2.16**	**1.52**	1	**1.52**	**2.88**	1	**2.88**
logistics98	6	6	6	6	**3.74**	1	3.71	**3.78**	1	3.75	**2.02**	1	**2.02**	**4.65**	1	**4.65**
miconic	141	141	141	141	0.82	**1.01**	0.83	0.99	1.02	**1.02**	1	1	1	1	1	1
mprime	**23**	20	22	20	0.74	**0.99**	0.75	1.03	0.99	**1.05**	**1.01**	1	**1.01**	**1.15**	1	**1.15**
mystery	15	15	15	15	0.63	0.98	0.64	1	0.98	**1.01**	1	1	1	**1.01**	1	**1.01**
openstacks	7	7	7	7	1	1	1	1	1	1	1	1	1	1	1	1
pathways-noneg	5	5	5	5	1.65	1.02	**1.7**	1.65	1.02	**1.7**	**1.31**	1	**1.31**	**1.87**	1	**1.87**
pipes-notankage	16	16	16	16	1.04	1.01	**1.05**	1.09	1.01	**1.1**	1	1	1	**1.09**	1	**1.09**
pipes-tankage	9	9	9	9	1	1	**1.01**	1.02	1	**1.04**	1	1	1	**1.04**	1	**1.04**
psr-small	49	49	49	49	1	**1.52**	0.99	1	**1.52**	1	1	**1.48**	1	1.08	**1.51**	1.08
rovers	7	**8**	7	**8**	2.61	1.04	**2.64**	2.63	1.04	**2.66**	**1.78**	1	**1.78**	3.23	1.02	**3.24**
satellite	7	**12**	7	**12**	13.74	1.22	**14.09**	14.55	1.22	**14.91**	**6.85**	1.04	**6.85**	19.18	1.35	**19.33**
tpp	6	6	6	6	1.03	1	**1.04**	**1.04**	1	**1.04**	**1.02**	1	**1.02**	**1.37**	1	**1.37**
transport	11	11	11	11	**1.5**	1	**1.5**	**1.54**	1	**1.54**	**1.24**	1	**1.24**	**1.79**	1	**1.79**
trucks	9	9	9	9	0.96	0.99	0.95	1	0.99	1	1	1	1	**1.01**	1	**1.01**
zenotravel	12	**13**	12	**13**	**2.7**	0.99	2.68	**2.81**	0.99	2.79	**1.54**	1	**1.54**	**2.86**	1	**2.86**
Total/Geometric Mean	441	**445**	440	**445**	1.339	1.047	**1.350**	1.424	1.048	**1.437**	1.226	1.023	**1.231**	1.558	1.055	**1.567**

Conclusion

- Three optimality and completess preserving prunining methods
- PP-A* guarantees optimality while using path-dependend pruning
- Partition-Based pruning effects more domains than Action Tunneling does
- The combined version shows best results

Thank you for your attention!