On an Attempt at Casting Orbit Search as a Task Transformation

Daniel Gnad[1] **David Speck**[1,2]

[1]Linköping University, Sweden

[2]University of Basel, Switzerland

LINKÖPING UNIVERSITY

University of Basel

Setting

Classical Planning – FDR with conditional effects

Setting

Classical Planning – FDR with conditional effects

Symmetry Breaking using Orbit Space Search

University of Basel

LINKÖPING UNIVERSITY

Setting

Classical Planning – FDR with conditional effects

Symmetry Breaking using Orbit Space Search

Research Question:
Instead of implementing a specialized algorithm,
can we encode symmetries into the task?

Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states.

Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)

$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.

Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

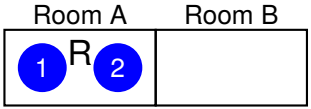
Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}$, $\mathcal{D}(r) = \{A, B\}$,
$\mathcal{D}(b_i) = \{A, B, R\}$



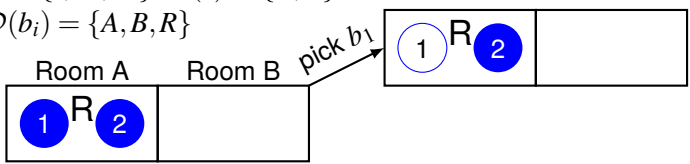
Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}$, $\mathcal{D}(r) = \{A, B\}$,
$\mathcal{D}(b_i) = \{A, B, R\}$



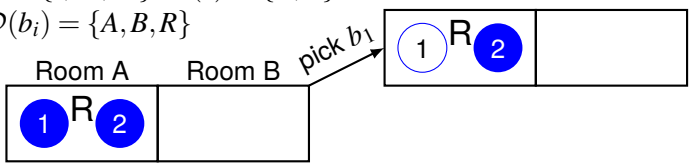
Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}$, $\mathcal{D}(r) = \{A, B\}$,
$\mathcal{D}(b_i) = \{A, B, R\}$



$r{=}A, b_1{=}R, b_2{=}A$

Symmetry Breaking

Orbit-Space Search

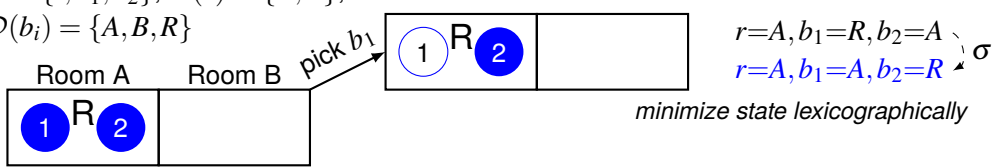
Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)

$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.

Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}, \mathcal{D}(r) = \{A, B\},$
$\mathcal{D}(b_i) = \{A, B, R\}$



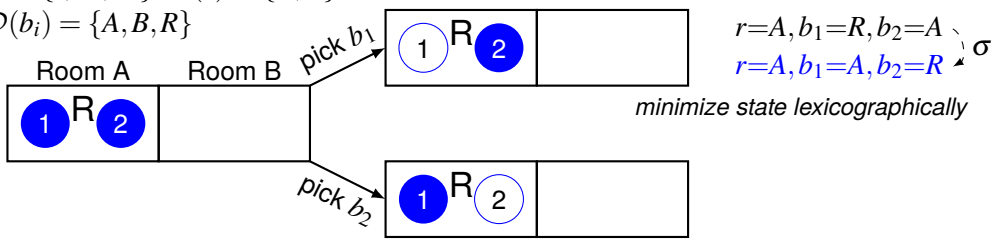
Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}, \mathcal{D}(r) = \{A, B\},$
$\mathcal{D}(b_i) = \{A, B, R\}$



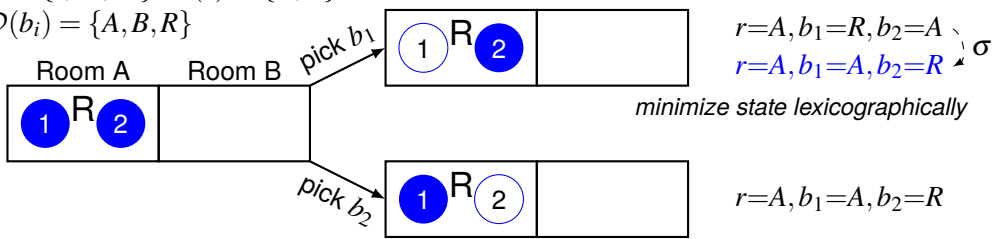
Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}$, $\mathcal{D}(r) = \{A, B\}$,
$\mathcal{D}(b_i) = \{A, B, R\}$



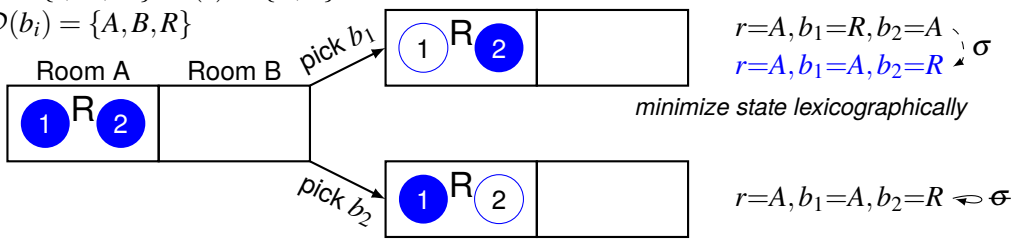
Symmetry Breaking

Orbit-Space Search

Prune states that are symmetric to previously seen states. (Pochter et al. 2011; Domshlak et al. 2015)
$\rightsquigarrow$ exponential reduction in search effort; preserves optimality.

Symmetries are permutations of facts σ that preserve problem structure.
Orbit Search: transform states to canonical representative.

$\mathcal{V} = \{r, b_1, b_2\}$, $\mathcal{D}(r) = \{A, B\}$,
$\mathcal{D}(b_i) = \{A, B, R\}$



Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.

Task Transformation: Encode application of permutation σ into conditional action effects.

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.

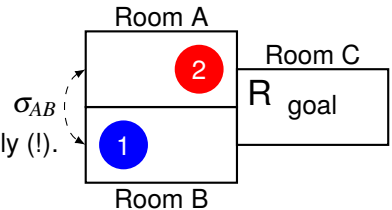
Task Transformation: Encode application of permutation σ into conditional action effects.

$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



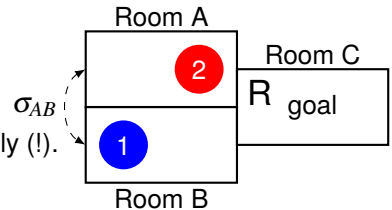
Task Transformation: Encode application of permutation σ into conditional action effects.

$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

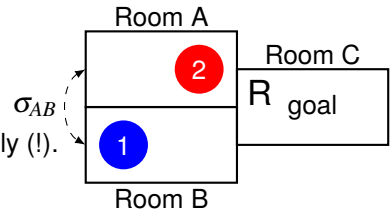
$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

$CBA \xrightarrow{a=\text{move}(C,B)} BBA$

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

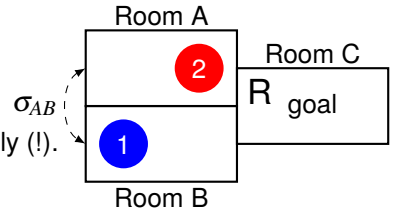
$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

$$CBA \xrightarrow{a=\text{move}(C,B)} BBA \xrightarrow{\sigma_{AB}} AAB$$

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

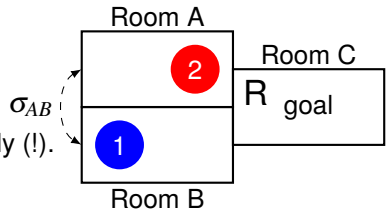
$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

$CBA \xrightarrow{a=\text{move}(C,B)} BBA \xrightarrow{\sigma_{AB}} AAB$ $\rightsquigarrow$ adapt $\mathit{eff}(a)[r] = A$, add $\mathit{eff}(a)[b_i] = \{A \rhd B, B \rhd A\}$

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

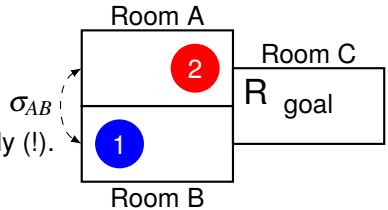
$CBA \xrightarrow{a=\text{move}(C,B)} BBA \xrightarrow{\sigma_{AB}} AAB$ $\rightsquigarrow$ adapt $\mathit{eff}(a)[r] = A$, add $\mathit{eff}(a)[b_i] = \{A \rhd B, B \rhd A\}$

transformed action: $CBA \xrightarrow{\text{move}(C,B)} AAB$ (preconditions / costs unchanged)

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

$CBA \xrightarrow{a=\text{move}(C,B)} BBA \xrightarrow{\sigma_{AB}} AAB$ $\rightsquigarrow$ adapt $\mathit{eff}(a)[r] = A$, add $\mathit{eff}(a)[b_i] = \{A \rhd B, B \rhd A\}$

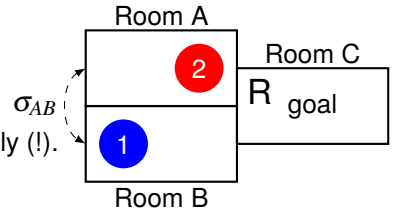
transformed action: $CBA \xrightarrow{\text{move}(C,B)} AAB$ (preconditions / costs unchanged)

How to decide on the permutation?

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

$CBA \xrightarrow{a=\mathrm{move}(C,B)} BBA \xrightarrow{\sigma_{AB}} AAB$ $\rightsquigarrow$ adapt $\mathit{eff}(a)[r] = A$, add $\mathit{eff}(a)[b_i] = \{A \rhd B, B \rhd A\}$

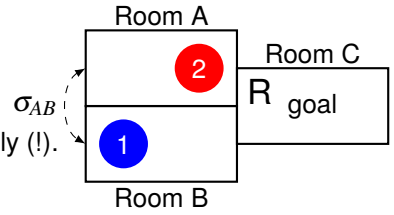
transformed action: $CBA \xrightarrow{\mathrm{move}(C,B)} AAB$ (preconditions / costs unchanged)

How to decide on the permutation? $\rightsquigarrow$ action post condition: $\mathrm{move} \rightarrow \underline{A}AA$.

Orbit Search as Task Transformation

Orbit search: expansion of a state s

1. Generate successors s' via actions a.
2. Find permutation σ that minimizes s' lexicographically (!).
3. Apply σ to s' and add $\sigma(s')$ to open list.



Task Transformation: Encode application of permutation σ into conditional action effects.

$\rightsquigarrow$ Fix permutation σ for each action at transformation time.

$CBA \xrightarrow{a=\text{move}(C,B)} BBA \xrightarrow{\sigma_{AB}} AAB$ $\rightsquigarrow$ adapt $\mathit{eff}(a)[r] = A$, add $\mathit{eff}(a)[b_i] = \{A \rhd B, B \rhd A\}$

transformed action: $CBA \xrightarrow{\text{move}(C,B)} AAB$ (preconditions / costs unchanged)

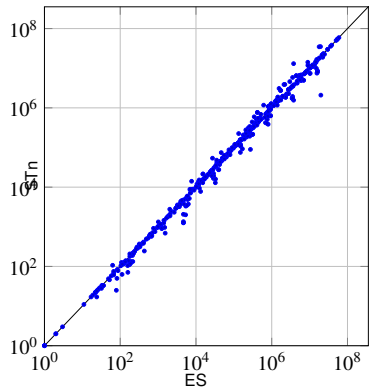
How to decide on the permutation? $\rightsquigarrow$ action post condition: $\text{move} \rightarrow \underline{A}AA$.

We are losing pruning potential! Gripper with N balls, pick action: $ABB\underline{B}BBB\ldots B$

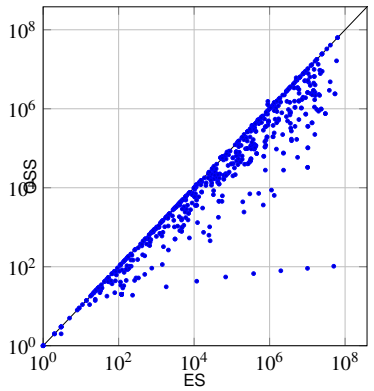
Experiments – Pruning Power

Expansions until last f layer

Baseline vs. Symmetry Transformation



Baseline vs. Orbit Search



Experiments – Coverage

Optimal Planning with Blind Heuristic

B: standard search
OSS: native orbit search

ST: symmetry transformation
n / **i** / **r**: fill up post condition with nothing / initial state / random state

Domain	#	B	OSS	STn	STi	STr
agricola	20	0	**5**	0	**5**	1
miconic	141	41	41	**45**	**45**	44
openstacks-08	30	22	**25**	22	**25**	23
openstacks-11	20	17	**19**	17	**19**	18
openstacks-14	14	2	**4**	2	**4**	2
pegsol-11	20	17	**18**	17	**18**	17
petri-net	20	**4**	2	**4**	**4**	3
psr-small	32	31	**32**	**32**	**32**	31
satellite	36	5	5	**6**	5	5
woodworking-08	22	2	2	2	**3**	2
woodworking-11	16	1	1	1	**2**	1
Sum	**1426**	517	**581**	515	508	483

Experiments – Coverage

Optimal Planning with Blind Heuristic

B: standard search
OSS: native orbit search

ST: symmetry transformation
n / **i** / **r**: fill up post condition with nothing / initial state / random state

Domain	#	B	OSS	STn	STi	STr
barman-11	20	4	**8**	4	4	4
depot	22	4	**5**	4	4	4
gripper	20	8	**20**	8	9	7
hiking-14	20	11	**14**	11	11	10
mprime	35	19	**20**	19	14	14
organic-split	20	10	**11**	10	10	9
pipesworld-T	50	12	**17**	10	9	8
scanalyzer-08	26	8	**10**	7	8	7
sokoban-08	27	19	**23**	19	16	17
tetris-14	17	**9**	**9**	8	5	5
zenotravel	19	**7**	**7**	6	6	6
Sum	**1426**	517	**581**	515	508	483

Conclusions

Summary

- Orbit search as task transformation.
- Reduction is limited by size of action post conditions.
- Per-state much faster than native orbit search.

Conclusions

Summary

- Orbit search as task transformation.
- Reduction is limited by size of action post conditions.
- Per-state much faster than native orbit search.

Future Work

- Context splitting on variables not in post condition.
- Support for conditional effects in input task.