Symbolic Search and Abstraction Heuristics for Cost-Optimal Planning

Álvaro Torralba
Advisors: Daniel Borrajo and Carlos Linares López

Universidad Carlos III de Madrid – June 2, 2015
1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Automated Planning

Given a **planning task:**

- A *logical description* of the *initial situation, goal condition* and a set of *possible actions*

\[
\begin{align*}
\mathcal{V} &= \{ \text{at-T} = \{A, B\}, \text{at-P} = \{A, B, T\} \} \\
\mathcal{s}_0 &= \{ \text{at-T A, at-P A} \} \\
\mathcal{s}_* &= \{ \text{at-P B} \} \\
\mathcal{O} &= \{ \text{move-T (A, B)}, \text{move-T (B, A)}, \text{load-P(A)}, \ldots \}
\end{align*}
\]

→ Find a **plan** (sequence of actions)
→ Cost-optimal: plan of minimum cost (prove it)
Empirical Evaluation Methods

SAS+ task → Planner → Optimal plan

→ Domain independent!! a planner can deal with any task
Empirical Evaluation Methods

→ Domain independent!! a planner can deal with any task

Empirical evaluation methods:

▶ Time limit: 30 minutes
▶ Memory limit: 4GB RAM
▶ Coverage: number of problems solved
▶ Time: solve problems faster
Motivation of this Thesis

- Improve state-of-the-art optimal planning
  - Efficiently solve optimal planning problems
- Techniques considered
  - Bidirectional search
  - Symbolic search
  - Abstraction heuristics
- Understand strengths/weaknesses
- Understand relation between techniques
Motivation of this Thesis

- Improve state-of-the-art optimal planning
  - Efficiently solve optimal planning problems
- Techniques considered
  - Bidirectional search
  - Symbolic search $\Rightarrow$ GAMER: winner of IPC 2008
  - Abstraction heuristics
    - $\Rightarrow$ Merge-and-shrink: runner-up and part of the winner of IPC 2011
- Understand strengths/weaknesses
- Understand relation between techniques
State of the Art in Cost-Optimal Planning

Explicit search

Symbolic search

Algorithms

$A^*$

Uniform-Cost

forward

backward

bidirectional

Heuristics

Delete-relaxation: $h^{\text{max}}$, $h^+$

Landmarks: $h^{\text{LA}}$, LM-cut

Abstractions: PDBs, M&S

Critical paths: $h^m$

Flow

Pruning techniques

State invariants

Symmetries

Partial-order pruning
State of the Art in Cost-Optimal Planning

Explicit search | Symbolic search

Algorithms

A* | Uniform-Cost

forward | backward | bidirectional

Heuristics

Delete-relaxation: \( h^{max} \), \( h^+ \)
Landmarks: \( h^{LA} \), LM-cut
Abstractions: PDBs, M&S
Critical paths: \( h^m \)
Flow  
  \[ \text{max} \] \[ \text{add} \] \[ \text{LP} \]

Pruning techniques

State invariants
Symmetries
Partial-order pruning
State of the Art in Cost-Optimal Planning

Explicit search
Symbolic search

Algorithms

A*
Uniform-Cost

forward
backward
bidirectional

Heuristics

Delete-relaxation: $h^{max}, h^+$
Landmarks: $h^{LA}, LM$-cut
Abstractions: PDBs, M&S
Critical paths: $h^m$
Flow

Pruning techniques

State invariants
Symmetries
Partial-order pruning
State of the Art in Cost-Optimal Planning

Explicit search

Symbolic search

Algorithms

- A* (forward)
- Uniform-Cost (backward, bidirectional)

Heuristics

- Delete-relaxation: $h^{max}, h^+$
- Landmarks: $h^{LA}$, LM-cut
- Abstractions: PDBs, M&S
- Critical paths: $h^m$
- Flow: max, add, LP

Pruning techniques

- State invariants
- Symmetries
- Partial-order pruning
State of the Art in Cost-Optimal Planning

Explicit search
Symbolic search

Algorithms
- $A^*$
- Uniform-Cost
- forward
- backward
- bidirectional

Heuristics
- Delete-relaxation: $h^{max}, h^+$
- Landmarks: $h^{LA}, LM-cut$
- Abstractions: PDBs, M&S
- Critical paths: $h^m$
- Flow
  - max
  - add
  - LP

Pruning techniques
- State invariants
- Symmetries
- Partial-order pruning
State of the Art in Cost-Optimal Planning

Explicit search

Symbolic search

Algorithms

A*  forward
Uniform-Cost  backward
bidirectional

Heuristics

Delete-relaxation: $h_{\text{max}}$, $h^+$
Landmarks: $h^{LA}$, LM-cut
Abstractions: PDBs, M&S
Critical paths: $h^m$
Flow

Pruning techniques

State invariants
Symmetries
Partial-order pruning
State of the Art in Cost-Optimal Planning

Explicit search

Symbolic search

Algorithms

A*
Uniform-Cost
forward
backward
bidirectional

Heuristics

Delete-relaxation: $h^{max}, h^+$
Landmarks: $h^{LA}, LM$-cut
Abstractions: PDBs, M&S
Critical paths: $h^m$
Flow

Pruning techniques

State invariants
Symmetries
Partial-order pruning
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
$S_0 \ • \ \ • \ S_*$
Reason with sets of states!

$S_g = 1$

$S_g = 2$

$S_0 \rightarrow S_1$

$S_0 \rightarrow S_2$

$S_0 \rightarrow S_3$

$S_0 \rightarrow S_4$

$S_\ast$
Reason with sets of states!

\[ S_\text{g} = 1 \]

\[ S_\ast \]
From Explicit to Symbolic Search

Reason with sets of states!

\[ S_{g=1}, S_{g=2} \]

- \( S_0 \)
- \( S_1 \) \( \cdots \) \( S_{10} \)
- \( S_{g=1} \)
- \( S_{g=2} \)
Binary Decision Diagrams (BDDs)

- Sets of states represented with Binary Decision Diagrams
  - Variable ordering
  - Reduction rules
- Possible exponential gain in memory/time
- Efficient operations (polynomial in BDD size)

1. (at Truck A) (at Package A)
2. (at Truck A) (in Package Truck)
3. (at Truck B) (at Package A)
Image Computation

- Expand a set of states and generate the successor states
- Transition Relation: BDD that represents one or more planning actions with the same cost

\[
S' \leftarrow \text{image}(S, T) = \exists x . \ S(x) \land T(x, x')[x' \leftrightarrow x]
\]

\[
S' \leftarrow \left(\text{move } T_1 \ A \ B\right) \ 
\left(\text{move } T_1 \ B \ A\right) \ 
\left(\text{load } P \ T_1 \ A\right) \ 
\ldots
\]
Symbolic Bidirectional Breadth-First Search

\[ g \quad 0 \quad s_0 \quad s_\ast \quad 0 \]
Symbolic Bidirectional Breadth-First Search

\[ S_0 \rightarrow S_1^g \rightarrow S_* \]

Decide forward or backward direction at each step.
Symbolic Bidirectional Breadth-First Search

- Decide forward or backward direction at each step

![Diagram showing symbolic bidirectional breadth-first search with states $S_0$, $S_1^g$, $S_1^h$, and $s_*$]
Symbolic Bidirectional Breadth-First Search

- Decide forward or backward direction at each step
Symbolic Bidirectional Breadth-First Search

- Decide forward or backward direction at each step

```
g 0 1 2 3
S_0 S_1^g S_2^g S_3^g
```

```
h 1 0
S_1^h s_*
```
Outline

1 Introduction
   - Cost-Optimal Planning

2 Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3 Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4 Symbolic Bidirectional Heuristic Search

5 Conclusions
   - Final Results: IPC14
   - Conclusions
Optimizing Image Computation

- Image computation is the main bottleneck in symbolic search
- How to represent the Transition Relation?
  - Monolithic relation $\Rightarrow$ may use exponential memory
  - Solution in GAMER $\Rightarrow$ One TR for each action

move-T (A, B)

load-P (A)

move-T (B, A)

\[ \ldots \]
Optimizing Image Computation

- Image computation is the main bottleneck in symbolic search
- How to represent the Transition Relation?
  - Monolithic relation ⇒ may use exponential memory
  - Solution in GAMER ⇒ One TR for each action
- Idea 1: Separate preconditions and effects
  → avoid using auxiliary variables!

```plaintext
move-T (A, B)     pre: at-T A     eff: at-T B
load-P (A)        pre: at-T A, at-P A eff: at-P T
move-T (B, A)     pre: at-T B     eff: at-T A
...```

Alvaro Torralba
PhD Defense
June 2, 2015 13 / 54
Optimizing Image Computation

- Image computation is the main bottleneck in symbolic search

How to represent the Transition Relation?
- Monolithic relation ⇒ may use exponential memory
- Solution in GAMER ⇒ One TR for each action

- Idea 1: Separate preconditions and effects
  → avoid using auxiliary variables!
- Idea 2: Conjunction Tree
  → check preconditions of all operators simultaneously

```
move-T (A, B)
load-P (A)
move-T (B, A)
...
```

```
/\ 
/ \  
/   \ 
/     \ 
at-P  
/   \  
/     \ 
T     
/ \  / \ 
/   /   \ 
A   B   * 
 eff: at-T A
```

```
/\ 
/ \  
/   \ 
/     \ 
eff: at-P T
/   \  
/     \ 
eff: at-P A
/ \  / \ 
/   /   \ 
/     \ 
eff: at-T B
```
Optimizing Image Computation

- Image computation is the main bottleneck in symbolic search
- How to represent the Transition Relation?
  - Monolithic relation \(\Rightarrow\) may use exponential memory
  - Solution in GAMER \(\Rightarrow\) One TR for each action
- Idea 1: Separate preconditions and effects
  \(\rightarrow\) avoid using auxiliary variables!
- Idea 2: Conjunction Tree
  \(\rightarrow\) check preconditions of all operators simultaneously
- Idea 3: Aggregate TRs
  \(\rightarrow\) different strategies to group the actions

```
moves-T (A, B)
load-P (A)
moves-T (B, A)

...```

```
moves-T (A, B)
load-P (A)
moves-T (B, A)```
Empirical Results

- Compare image computation methods:
  1. $TR^1$: baseline approach
  2. $TR^{1+}$: avoid using $x'$ variables
  3. $CT^L_{20}$: conjunction tree
  4. $T^{DT}_{100k}$: aggregate TRs

Total coverage of symbolic search algorithms over 1375 instances:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$TR^1$</th>
<th>$TR^{1+}$</th>
<th>$CT^L_{20}$</th>
<th>$T^{DT}_{100k}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward uniform-cost search</td>
<td>699</td>
<td>676</td>
<td>724</td>
<td>742</td>
</tr>
<tr>
<td>Backward uniform-cost search</td>
<td>444</td>
<td>525</td>
<td>529</td>
<td>532</td>
</tr>
<tr>
<td>Bidirectional uniform-cost search</td>
<td>729</td>
<td>763</td>
<td>769</td>
<td>793</td>
</tr>
<tr>
<td>BDDA* with SPDBs</td>
<td>705</td>
<td>717</td>
<td>724</td>
<td>764</td>
</tr>
</tbody>
</table>

$TR^1 \leq TR^{1+} \leq CT^L_{20} \leq T^{DT}_{100k}$
( across all domains)
Time of Bidirectional Search

Solving time of $T_{100}^{DT}$ (seconds)

Solving time of $TR^1$ (seconds)
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Motivation: State Invariants in Symbolic Search

- Invariant: holds in all states that may belong to a solution path
  1. Mutex: pair of facts that cannot be true in the same state
     → a truck cannot be simultaneously at two locations
  2. Invariant group: Set of facts such that exactly one is true
     → a truck must be somewhere

- Generated computing $h^2$ in both directions
- Useful for:
  1. Removing operators from the planning task
  2. Pruning invalid states during the search
Encoding Constraints with \textit{cBDD}

- \textit{cBDD}: BDD that describes invalid states
  - 1. Mutex: $f_i \land f_j$
  - 2. “At-least-1” invariant: $\neg (f_1 \lor f_2 \lor \ldots \lor f_n)$
- Remove invalid states from $S_g$: $S_g \setminus cBDD$

\textit{e-deletion}: encode invariants in the TRs

\[ \rightarrow \text{no invalid states are generated} \]
Experimental Results

- Constraints found in 35 out of 43 domains
- 10%-74% invalid operators found in 17 out of 43 domains
- Mutex types:
  - Baseline (B)
  - Not pruning invalid states: $\mathcal{M}_\emptyset$
  - Pruning invalid states: $cBDD$ or $e$-deletion ($e$-del)

<table>
<thead>
<tr>
<th>Search Method</th>
<th>B</th>
<th>$\mathcal{M}_\emptyset$</th>
<th>cBDD</th>
<th>e-del</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward uniform-cost search</td>
<td>699</td>
<td>742</td>
<td>745</td>
<td>750</td>
</tr>
<tr>
<td>Backward uniform-cost search</td>
<td>509</td>
<td>532</td>
<td>677</td>
<td>696</td>
</tr>
<tr>
<td><strong>Bidirectional uniform-cost search</strong></td>
<td>765</td>
<td>793</td>
<td>836</td>
<td><strong>841</strong></td>
</tr>
<tr>
<td>BDDA* with SPDBs</td>
<td>736</td>
<td>764</td>
<td>777</td>
<td>781</td>
</tr>
</tbody>
</table>
Time of Bidirectional Uniform-Cost Search

(f) remove operators

(g) prune invalid states

(h) e-deletion vs cBDD
Comparison with State-of-the-Art Planners

![Graph comparing coverage over time for different planners]

- **CGAMER-BD**
- **GAMER-BD**

Coverage

Time (seconds)
Comparison with State-of-the-Art Planners

![Graph showing comparison of State-of-the-Art Planners]

- **CGAMER-FW**
- **EXPLICIT-FW**

**Axes:**
- **Y-axis:** Coverage
- **X-axis:** Time (seconds)

Legend:
- Red dotted line: CGAMER-FW
- Black dotted line: EXPLICIT-FW
Comparison with State-of-the-Art Planners

![Graph showing comparison between different planners.](image-url)
Comparison with State-of-the-Art Planners

![Graph showing comparison with state-of-the-art planners](image.png)

- CGAMER-BD
- GAMER-BD
- CGAMER-FW
- EXPLICIT-FW
- A* + LM-CUT
- BDDA* + SPDBs
Summary

1. Image computation
   - Analyzed different methods for image computation
   - Best method: aggregate TRs

2. State invariants
   - Pruning invalid states (specially useful in bw search)
   - Best encoding for symbolic search: e-edeletion

These significantly improved performance of symbolic planning
→ Symbolic bidirectional blind search is the current state-of-the-art for cost-optimal planning
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Motivation: Heuristics in Symbolic Search

Heuristics

- **Delete-relaxation**: $h_{max}, h^+$
- **Landmarks**: $h^{LA}$, LM-cut
- **Abstractions**: PDBs, M&S, CEGAR, Fork
- **Critical paths**: $h^m$
- **Flow**
  - $h_{max}$
  - add
  - LP
Motivation: Heuristics in Symbolic Search

Heuristics

- Delete-relaxation: $h^{max}$, $h^+$
- Landmarks: $h^{LA}$, LM-cut
- Abstractions: PDBs, M&S, CEGAR, Fork
- Critical paths: $h^m$
- Flow

Challenge: How to evaluate $h(s)$ on a set of states?
Motivation: Heuristics in Symbolic Search

<table>
<thead>
<tr>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete-relaxation:  $h^{\text{max}}$, $h^+$</td>
</tr>
<tr>
<td>Landmarks:          $h^{LA}$, LM-cut</td>
</tr>
<tr>
<td>Abstractions:       PDBs, M&amp;S, CEGAR, Fork</td>
</tr>
<tr>
<td>Critical paths:     $h^m$</td>
</tr>
<tr>
<td>Flow</td>
</tr>
<tr>
<td>max</td>
</tr>
</tbody>
</table>

Challenge: How to evaluate $h(s)$ on a set of states?
Abstraction Heuristics

- Abstraction: Mapping from states to abstract states
  - Smaller abstract state space → easier to search
  - Use optimal distances in abstract state space as heuristic
  - Preserve transitions → admissible estimation

```
00  4  5  6  7
  01
  02
start

01  3  2  1  0
  11
  12

10  2  1  0

20  1

30
```

Pattern Databases (PDBs)
- Ignore some variables in the problem
- Limitation: ignoring a single variable may relax too much
Abstraction Heuristics

- Abstraction: Mapping from states to abstract states
  - Smaller abstract state space → easier to search
  - Use optimal distances in abstract state space as heuristic
  - Preserve transitions → admissible estimation

Pattern Databases (PDBs)
- Ignore some variables in the problem
- Limitation: ignoring a single variable may relax too much
Abstraction Heuristics

- **Abstraction**: Mapping from states to abstract states
  - Smaller abstract state space → easier to search
  - Use optimal distances in abstract state space as heuristic
  - Preserve transitions → admissible estimation

**Pattern Databases (PDBs)**
- Ignore some variables in the problem
- Limitation: ignoring a single variable may relax too much
Abstraction Heuristics

- Abstraction: Mapping from states to abstract states
  - Smaller abstract state space → easier to search
  - Use optimal distances in abstract state space as heuristic
  - Preserve transitions → admissible estimation

Pattern Databases (PDBs)
- Ignore some variables in the problem
- Limitation: ignoring a single variable may relax too much
Merge-and-Shrink Algorithm (M&S)

Algorithm 1: M&S

\[
\alpha_1 \leftarrow \Pi_{v_1}
\]

foreach \( v \in \{ v_2 \ldots v_n \} \):

if \( |\alpha| > N \):

    shrink(\( \alpha_{i-1} \)) \otimes \Pi_i

\( \alpha_i \leftarrow \alpha_{i-1} \otimes \Pi_i \)

return \( \alpha \)

- Merge strategy: **Linear**
  - variable ordering
- Shrink strategy
  - reduce abstraction size
Merge-and-Shrink Algorithm (M&S)

Algorithm 1: M&S

\[ \alpha_1 \leftarrow \Pi \nu_1 \]

foreach \( \nu \in \{\nu_2 \ldots \nu_n\} \):

if \( |\alpha| > N \):

shrinks \( (\alpha_{i-1} \land \Pi_i) \)

\[ \alpha_i \leftarrow \alpha_{i-1} \land \Pi_i \]

return \( \alpha \)

- Merge strategy: **Linear**
  - variable ordering
- Shrink strategy
  - reduce abstraction size

\[ \alpha_1 = T_A \quad \text{move}_{A,B} \]
\[ \text{move}_{B,A} \]
Merge-and-Shrink Algorithm (M&S)

**Algorithm 1: M&S**

\[ \alpha_1 \leftarrow \Pi_{\nu_1} \]

foreach \( \nu \in \{\nu_2 \ldots \nu_n\} \):

if |\( \alpha \)| > \( N \):

\[ \text{shrink}(\alpha_{i-1}) \otimes \Pi_i \]

\( \alpha_i \leftarrow \alpha_{i-1} \otimes \Pi_i \)

return \( \alpha \)

- **Merge strategy:** Linear
  - \( \rightarrow \) variable ordering
- **Shrink strategy**
  - \( \rightarrow \) reduce abstraction size
Algorithm 1: M&S

\[ \alpha_1 \leftarrow \Pi_{\nu_1} \]

\textbf{foreach} \( \nu \in \{\nu_2 \ldots \nu_n\} : \)

\textbf{if} \( |\alpha| > N : \)

\textbf{shrink} \((\alpha_{i-1}) \otimes \Pi_i \)

\[ \alpha_i \leftarrow \alpha_{i-1} \otimes \Pi_i \]

\textbf{return} \( \alpha \)

- Merge strategy: \textbf{Linear} \rightarrow \text{variable ordering}
- Shrink strategy \rightarrow \text{reduce abstraction size}
Outline

1 Introduction
   - Cost-Optimal Planning

2 Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3 Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4 Symbolic Bidirectional Heuristic Search

5 Conclusions
   - Final Results: IPC14
   - Conclusions
Hypothesis: BDDA* lacks good heuristics
→ BDDA* + M&S can improve results
How to use M&S in symbolic search:

M&S algorithm

M&S heuristic to ADD

ADD to BDDs

BDDs to use in symbolic search
Merge-and-Shrink as ADDs

\[ \alpha_1 = T_A \]

move\_A,B

\[ \alpha_3 = T_A, P_T, P_A \]

move\_A,B

move\_B,A

load/unload\_A

move\_A,B

move\_A,B

load/unload\_B:
Merge-and-Shrink as ADDs

\[ \alpha_1 = T_A \]

\[ \text{move}_{A,B} \]

\[ \begin{array}{c}
\text{A} \\
\text{B}
\end{array} \]

\[ \text{move}_{B,A} \]

\[ \alpha_3 = T_A, P_T, P_A \]

\[ \begin{array}{c}
\text{AA} \\
\text{BA}
\end{array} \]

\[ \text{move}_{A,B} \]

\[ \text{load/unload}_A \]

\[ \begin{array}{c}
\text{Ac} \\
\text{Bc}
\end{array} \]

\[ \text{move}_{B,A} \]

\[ \text{move}_{A,B} \]

\[ \text{load/unload}_B \]

\[ \begin{array}{c}
\text{AB} \\
\text{BB}
\end{array} \]

\[ \text{move}_{B,A} \]
Merge-and-Shrink as ADDs

\[ \alpha_1 = T_A \]
move_{A,B}

\[ \alpha_3 = T_A, P_T, P_A \]
move_{A,B}

load/unload_{A}

move_{B,A}

move_{A,B}

load/unload_{B}
Merge-and-Shrink as ADDs

\[ \alpha_1 = T_A \]

- move\(_{A,B}^\text{move}\)
- move\(_{B,A}^\text{move}\)

\[ \alpha_3 = T_A, P_T, P_A \]

- load/unload\(_A^\text{load/unload}\)
- move\(_{B,A}^\text{move}\)
- move\(_{A,B}^\text{move}\)
- move\(_{B,A}^\text{move}\)

- load/unload\(_B^\text{load/unload}\)
- move\(_{A,B}^\text{move}\)
- move\(_{B,A}^\text{move}\)
Merge-and-Shrink as ADDs

\[ \alpha_1 = T_A \]

\[ \text{move}_{A,B} \]

\[ \alpha_3 = T_A, P_T, P_A \]

\[ \text{move}_{A,B} \]

\[ \text{load/unload}_A \]

\[ \text{move}_{B,A} \]

\[ \text{move}_{A,B} \]

\[ \text{load/unload}_B \]
Merge-and-Shrink as ADDs

\[ \alpha_1 = T_A \]

move_{A,B}

\[ \alpha_3 = T_A, P_T, P_A \]

move_{A,B}

load/unload_A

move_{B,A}

move_{A,B}

load/unload_B:

load/unload_B:
Theoretical Results

- M&S to ADDs/BDDs in polynomial time
- Related empirical results:
  - ADD representation of heuristics reduces memory
  - Variable ordering has a huge impact

- ADD/BDD reduction rules may achieve exponential gain in memory with respect to shrinking perfect strategies
  - shows potential of improvement for M&S strategies
Empirical Results

- Used M&S in symbolic search $\rightarrow$ Worse than symbolic PDBs

Contradicts our hypothesis
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Motivation: Combine Symbolic Search and M&S

1. Symbolic PDBs: larger abstract state spaces
2. M&S: flexible abstractions

Can we get the best of both worlds?
Motivation: Combine Symbolic Search and M&S

1. Symbolic PDBs: larger abstract state spaces
2. M&S: flexible abstractions

Can we get the best of both worlds?

→ Use symbolic search to search M&S abstractions!

Symbolic Perimeter M&S:

1. Symbolic M&S abstractions: larger M&S abstract state spaces
2. Perimeter abstractions
SM&S Hierarchy

Enlarged M&S abstractions: to perform symbolic search
SM&S Hierarchy

Enlarged M&S abstractions: to perform symbolic search

\[ \alpha_0^{\text{SM&S}} \] (original problem)
\[ \alpha_1^{\text{SM&S}} \]
\[ \alpha_2^{\text{SM&S}} \]
\[ \alpha_3^{\text{SM&S}} \] (M&S abstraction)
\[ \alpha_4^{\text{SM&S}} \]
SM&S Hierarchy

Enlarged M&S abstractions: to perform symbolic search

\[ \alpha_0^{\text{SM&S}} \quad \text{(original problem)} \]
\[ \alpha_1^{\text{SM&S}} \]
\[ \alpha_2^{\text{SM&S}} \]
\[ \alpha_3^{\text{SM&S}} \]
\[ \alpha_4^{\text{SM&S}} \quad \text{(M&S abstraction)} \]
SM&S Hierarchy

Enlarged M&S abstractions: to perform symbolic search

\[ V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow V_4 \rightarrow V_5 \]

\[ \alpha_1 \rightarrow \alpha_2 \rightarrow \alpha_3 \rightarrow \alpha_4 \]

\[ \alpha_0^{\text{SM&S}} \] (original problem)

\[ \alpha_1^{\text{SM&S}} \]

\[ \alpha_2^{\text{SM&S}} \]

\[ \alpha_3^{\text{SM&S}} \]

\[ \alpha_4^{\text{SM&S}} \] (M&S abstraction)
SM&S Hierarchy

Enlarged M&S abstractions: to perform symbolic search

$\alpha_1 \rightarrow \alpha_2 \rightarrow \alpha_3 \rightarrow \alpha_4 \rightarrow \alpha_5$

$\alpha_0^{\text{SM&S}}$ (original problem)

$\alpha_1^{\text{SM&S}}$

$\alpha_2^{\text{SM&S}}$

$\alpha_3^{\text{SM&S}}$

$\alpha_4^{\text{SM&S}}$ (M&S abstraction)
SM&S Hierarchy

Enlarged M&S abstractions: to perform symbolic search

\[ \alpha_1 \xrightarrow{V_1} \alpha_2 \xrightarrow{V_3} \alpha_3 \xrightarrow{V_4} \alpha_4 \xrightarrow{V_5} \alpha_{SM&S}^0 \quad \text{(original problem)} \]

\[ \alpha_{SM&S}^1 \quad \alpha_{SM&S}^2 \quad \alpha_{SM&S}^3 \quad \alpha_{SM&S}^4 \quad \text{(M&S abstraction)} \]
Perimeter Abstractions

Challenges addressed with symbolic search

1. Regression
2. Expensive operations:
   - membership in perimeter
   - frontier mapping
3. Set perimeter radius

Contributions

1. Multiple abstraction levels
2. Improved initialization of abstract searches
Perimeter Abstractions

- Challenges addressed with symbolic search
  1. Regression
  2. Expensive operations:
     • membership in perimeter
     • frontier mapping
  3. Set perimeter radius

- Contributions
  1. Multiple abstraction levels
  2. Improved initialization of abstract searches

\[ S_\star \quad \text{Exp}(\alpha_0) \quad \text{Exp}(\alpha_1) \quad \text{Exp}(\alpha_2) \]
\[ V_{\alpha_0} = \emptyset \]

The Symbolic Perimeter Merge-and-Shrink (SPM&S) heuristic is admissible and consistent.
Symbolic Perimeter Merge-and-Shrink

\[ \mathcal{V}_{\alpha_0} = \emptyset \quad \mathcal{V}_{\alpha_1} \]

**M&S**

\[ \exp(\alpha_0) \]

- \[ h_\exp(\alpha_0) \]

- \[ h_{\text{SPM&S}} \] heuristic is admissible and consistent
Symbolic Perimeter Merge-and-Shrink

\[ V_{\alpha_0} = \emptyset \quad V_{\alpha_1} \]

M&S

Exp(\( \alpha_0 \))

\[ S_{*,0} \rightarrow S_{1,1} \rightarrow S_{2,2} \]

\[ h_{\text{Exp}(\alpha_0)} \]

truncated

Exp(\( \alpha_1 \))

\[ \alpha_1 \]

\[ S_{2,2} \rightarrow S_{3,3} \]

\[ h_{\text{Exp}(\alpha_1)} \]

truncated

\( h_{\text{SPM&S}} \) heuristic is admissible and consistent
Symbolic Perimeter Merge-and-Shrink

$h_{\text{SPM&S}}$ heuristic is admissible and consistent.
Empirical Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM-CUT</td>
<td>796</td>
</tr>
<tr>
<td>M&amp;S</td>
<td>650</td>
</tr>
<tr>
<td>SPM&amp;S</td>
<td>809</td>
</tr>
<tr>
<td>SPPDB</td>
<td>822</td>
</tr>
<tr>
<td>SP</td>
<td>800</td>
</tr>
</tbody>
</table>
Empirical Results: Expanded Nodes

Expanded nodes M&S bop 10k

Expanded nodes SPM&S bop 10k

Expanded nodes LM-CUT

Expanded nodes SPM&S bop 10k
Empirical Results: Expanded Nodes

Expanded nodes SPM&S bop 10k

Expanded nodes SP

Expanded nodes SPPDB
Summary

Symbolic Perimeter M&S
- Combines M&S, perimeter abstractions and symbolic search
- Improvements to perimeter abstractions
- Synergy between symbolic search and perimeter abstractions
- More accurate heuristic than both!

But...

Results still worse than symbolic bidirectional uniform-cost search
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Motivation: Heuristics in Symbolic Bidirectional Search

Observations

1. Bidirectional brute-force search is a state-of-the-art technique
2. Good symbolic abstraction heuristics
Motivation: Heuristics in Symbolic Bidirectional Search

- Observations
  1. Bidirectional brute-force search is a state-of-the-art technique
  2. Good symbolic abstraction heuristics

- Use abstraction heuristics in symbolic bidirectional search!
Motivation: Heuristics in Symbolic Bidirectional Search

Observations

1. Bidirectional brute-force search is a state-of-the-art technique
2. Good symbolic abstraction heuristics

Use abstraction heuristics in symbolic bidirectional search!

However, bidirectional heuristic search is not so easy:

- Very promising since years ago
- Never really able to outperform A* or bidirectional uniform-cost search
Algorithm

- Main idea:
  1. Start symbolic bidirectional uniform-cost search
     - If it succeeds → done!
  2. Detect when it is going to fail and activate heuristics

- Abstraction heuristics: Bidirectional, Partial, Perimeter
Algorithm

- **Main idea:**
  1. Start symbolic bidirectional uniform-cost search
     - If it succeeds → done!
  2. Detect when it is going to fail and activate heuristics
- **Abstraction heuristics:** Bidirectional, Partial, Perimeter
- Decide which search advance: **useful** and **feasible**
Main idea:
1. Start symbolic bidirectional uniform-cost search
   ✫ If it succeeds → done!
2. Detect when it is going to fail and activate heuristics

Abstraction heuristics: Bidirectional, Partial, Perimeter

Decide which search advance: **useful** and **feasible**
Algorithm

- **Main idea:**
  1. Start symbolic bidirectional uniform-cost search
     - If it succeeds → done!
  2. Detect when it is going to fail and activate heuristics

- Abstraction heuristics: Bidirectional, Partial, Perimeter
- Decide which search advance: **useful** and **feasible**
Algorithm

- **Main idea:**
  1. Start symbolic bidirectional uniform-cost search
     - If it succeeds → done!
  2. Detect when it is going to fail and activate heuristics

- Abstraction heuristics: Bidirectional, Partial, Perimeter
- Decide which search advance: **useful** and **feasible**
Empirical Results

- **Best**: 873
- **SM&S**: 840
- **PDB\textsubscript{ran}**: 844
- **PDB\textsubscript{cgl}**: 842
- **BD (\emptyset)**: 842

Coverage

Full SymBA*
Empirical Results

- **Best**
  - Full SymBA*: 870
  - No perimeter abstraction: 873

- **SM&S**
  - Full SymBA*: 840
  - No perimeter abstraction: 842

- **PDB_{ran}**
  - Full SymBA*: 844
  - No perimeter abstraction: 836

- **PDB_{cgl}**
  - Full SymBA*: 844
  - No perimeter abstraction: 842

- **BD (∅)**
  - Full SymBA*: 842
  - No perimeter abstraction: 842

**Coverage**

- 820
- 840
- 860
- 880
Empirical Results

Bar chart showing coverage for different datasets and abstraction methods:
- **Best** with coverage of 870 and 873.
- **SM&S** with coverage of 842.
- **PDB\textsubscript{ran}** with coverage of 836.
- **PDB\textsubscript{cgl}** with coverage of 837.
- **BD (\emptyset)** with coverage of 842.

Legend:
- Full SymBA\textsuperscript{*}
- No perimeter abstraction
- No bidir abstraction
Summary

Contributions:
- SymBA*: a symbolic bidirectional heuristic search algorithm
- Bidirectional search in abstract state spaces
- Synergy: Symbolic search + Bidirectional search + Perimeter abstractions

Symbolic Bidirectional A* is possible
- Future work: domain-independent abstraction strategies (better than a random selection)
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Final Results

![Graph showing coverage over time for Sym-BD and GAMER-BD](image)

- Sym-BD
- GAMER-BD

Time (seconds)

Coverage

- 10
- 200
- 400
- 600
- 800
Final Results

![Graph showing coverage over time for Sym-BD, GAMER-BD, and SymBA*]
Final Results

![Graph showing coverage over time for different methods: SPPDBmulti, SPPDB, SP, LM-CUT, M&S\(^b\).]
Final Results

![Graph showing the coverage over time for different methods.](image)

- Sym-BD
- GAMER-BD
- SymBA*
- SPPDBmulti
- SPPDB
- SP
- LM-CUT
- M&S

**Axes:**
- Y-axis: Coverage
- X-axis: Time (seconds)

**Legend:**
- Red dashed line: Sym-BD
- Blue dashed line: GAMER-BD
- Black dashed line: SymBA*
- Purple dotted line: SPPDBmulti
- Green dotted line: SPPDB
- Orange line: SP
- Cyan dotted line: LM-CUT
- Pink dotted line: M&S

**PhD Defense**
- June 2, 2015
2014 International Planning Competition

- Submitted our planners to the 2014-IPC
  1. CGAMER: Symbolic Bidirectional uniform-cost search with image computation and state-invariant constraints
  2. SPM&S: A* with symbolic perimeter PDBs and M&S

- Competed against:
  - GAMER: baseline symbolic planner
  - Top explicit-state search planners and portfolios

- Disclaimer: IPC results are not everything
  - Domains/Instances selection, bugs, ...
### 2014 International Planning Competition

<table>
<thead>
<tr>
<th></th>
<th>Barman</th>
<th>Cave</th>
<th>Childsnack</th>
<th>Citycar</th>
<th>Flootile</th>
<th>GED</th>
<th>Hiking</th>
<th>Maintenan</th>
<th>Openstack</th>
<th>Parking</th>
<th>Tetris</th>
<th>Tidybot</th>
<th>Transport</th>
<th>Visitall</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>symba-2</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>20</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td></td>
<td>151</td>
</tr>
<tr>
<td>symba-1</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>19</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>4</td>
<td>9</td>
<td>6</td>
<td></td>
<td>143</td>
</tr>
<tr>
<td>cgamer-bd</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>18</td>
<td>20</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>19</td>
<td>3</td>
<td>11</td>
<td>13</td>
<td>8</td>
<td>6</td>
<td>120</td>
</tr>
<tr>
<td>spmas</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>20</td>
<td>18</td>
<td>12</td>
<td>4</td>
<td>14</td>
<td>4</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>114</td>
</tr>
<tr>
<td>rida</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>16</td>
<td>5</td>
<td>19</td>
<td>17</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>15</td>
<td>113</td>
</tr>
<tr>
<td>dynamic-gamer</td>
<td>3</td>
<td>3</td>
<td>10</td>
<td>15</td>
<td>14</td>
<td>0</td>
<td>17</td>
<td>3</td>
<td>19</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>99</td>
</tr>
<tr>
<td>all-paca</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>6</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>12</td>
<td>98</td>
</tr>
<tr>
<td>cedalion</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>14</td>
<td>5</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>93</td>
</tr>
<tr>
<td>metis</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>91</td>
</tr>
<tr>
<td>nucler</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>0</td>
<td>7</td>
<td>13</td>
<td>90</td>
</tr>
<tr>
<td>rlazya</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>88</td>
</tr>
<tr>
<td>gamer</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>18</td>
<td>13</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>5</td>
<td>83</td>
</tr>
<tr>
<td>hflow</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td></td>
<td>53</td>
</tr>
<tr>
<td>miplan</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td>47</td>
</tr>
<tr>
<td>dpmpplan</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td>43</td>
</tr>
<tr>
<td>hpp-ce</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>hpp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
</tr>
</tbody>
</table>
## 2014 International Planning Competition

<table>
<thead>
<tr>
<th></th>
<th>Barman</th>
<th>Cave</th>
<th>Childsnacl</th>
<th>Citycar</th>
<th>Floorile</th>
<th>GED</th>
<th>Hiking</th>
<th>Maintenan</th>
<th>Openstack</th>
<th>Parking</th>
<th>Tetris</th>
<th>Tidybot</th>
<th>Transport</th>
<th>Visitall</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>symba-2</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>151</td>
<td></td>
<td></td>
</tr>
<tr>
<td>symba-1</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>19</td>
<td>20</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>4</td>
<td>9</td>
<td>6</td>
<td>143</td>
</tr>
<tr>
<td>cgamer-bd</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>18</td>
<td>20</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>19</td>
<td>3</td>
<td>11</td>
<td>13</td>
<td>8</td>
<td>6</td>
<td>120</td>
</tr>
<tr>
<td>spmas</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>20</td>
<td>18</td>
<td>12</td>
<td>4</td>
<td>14</td>
<td>4</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>114</td>
</tr>
<tr>
<td>rida</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>16</td>
<td>5</td>
<td>19</td>
<td>17</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>15</td>
<td>113</td>
</tr>
<tr>
<td>dynamic-gamer</td>
<td>3</td>
<td>3</td>
<td>10</td>
<td>15</td>
<td>14</td>
<td>0</td>
<td>17</td>
<td>3</td>
<td>19</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>99</td>
</tr>
<tr>
<td>all-paca</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>6</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>12</td>
<td>98</td>
</tr>
<tr>
<td>cedalion</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>14</td>
<td>5</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>93</td>
</tr>
<tr>
<td>metis</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>91</td>
</tr>
<tr>
<td>nucelar</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>0</td>
<td>7</td>
<td>13</td>
<td>90</td>
</tr>
<tr>
<td>rlazya</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>88</td>
</tr>
<tr>
<td>gamer</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>18</td>
<td>13</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>5</td>
<td>83</td>
</tr>
<tr>
<td>hflow</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>53</td>
</tr>
<tr>
<td>miplan</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td>47</td>
</tr>
<tr>
<td>dpmpplan</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>hpp-ce</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>hpp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
</tr>
</tbody>
</table>
## 2014 International Planning Competition

<table>
<thead>
<tr>
<th></th>
<th>Barman</th>
<th>Cave</th>
<th>Childsnack</th>
<th>Citycar</th>
<th>Floortile</th>
<th>GED</th>
<th>Hiking</th>
<th>Maintainan</th>
<th>Openstack</th>
<th>Parking</th>
<th>Tetris</th>
<th>Tidybot</th>
<th>Transport</th>
<th>Visitall</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>symba-2</strong></td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>151</td>
</tr>
<tr>
<td><strong>symba-1</strong></td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>19</td>
<td>20</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>4</td>
<td>9</td>
<td>6</td>
<td>143</td>
</tr>
<tr>
<td><strong>cgamer-bd</strong></td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>18</td>
<td>20</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>19</td>
<td>3</td>
<td>11</td>
<td>13</td>
<td>8</td>
<td>6</td>
<td>120</td>
</tr>
<tr>
<td><strong>spmas</strong></td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>20</td>
<td>18</td>
<td>12</td>
<td>4</td>
<td>14</td>
<td>4</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>114</td>
</tr>
<tr>
<td><strong>rida</strong></td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>16</td>
<td>5</td>
<td>19</td>
<td>17</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>15</td>
<td>113</td>
</tr>
<tr>
<td><strong>dynamic-gamer</strong></td>
<td>3</td>
<td>3</td>
<td><strong>10</strong></td>
<td>15</td>
<td>14</td>
<td>0</td>
<td>17</td>
<td>3</td>
<td>19</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>99</td>
</tr>
<tr>
<td><strong>all-paca</strong></td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>6</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>12</td>
<td>98</td>
</tr>
<tr>
<td><strong>cedalion</strong></td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>14</td>
<td>5</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>93</td>
</tr>
<tr>
<td><strong>metis</strong></td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>91</td>
</tr>
<tr>
<td><strong>nucelar</strong></td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>0</td>
<td>7</td>
<td>13</td>
<td>90</td>
</tr>
<tr>
<td><strong>rlazya</strong></td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>88</td>
</tr>
<tr>
<td><strong>gamer</strong></td>
<td>3</td>
<td>3</td>
<td>2</td>
<td><strong>18</strong></td>
<td>13</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>5</td>
<td>83</td>
</tr>
<tr>
<td><strong>hflow</strong></td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td><strong>15</strong></td>
<td>53</td>
</tr>
<tr>
<td><strong>miplan</strong></td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td>47</td>
</tr>
<tr>
<td><strong>dpmplan</strong></td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td><strong>hpp-ce</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td><strong>hpp</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>14</td>
</tr>
</tbody>
</table>
# 2014 International Planning Competition

<table>
<thead>
<tr>
<th></th>
<th>Barman</th>
<th>Cave</th>
<th>Childsnack</th>
<th>Citycar</th>
<th>Floortile</th>
<th>GED</th>
<th>Hiking</th>
<th>Maintainan</th>
<th>Openstack</th>
<th>Parking</th>
<th>Tetris</th>
<th>Tidybot</th>
<th>Transport</th>
<th>Visitall</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>symba-2</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>20</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td></td>
<td>151</td>
</tr>
<tr>
<td>symba-1</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>18</td>
<td>20</td>
<td>19</td>
<td>4</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>4</td>
<td>9</td>
<td>6</td>
<td></td>
<td>143</td>
</tr>
<tr>
<td>cgamer-bd</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>18</td>
<td>20</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>19</td>
<td>3</td>
<td>11</td>
<td>13</td>
<td>8</td>
<td>6</td>
<td>120</td>
</tr>
<tr>
<td>spmas</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>20</td>
<td>18</td>
<td>12</td>
<td>4</td>
<td>14</td>
<td>4</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>114</td>
</tr>
<tr>
<td>rida</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>16</td>
<td>5</td>
<td>19</td>
<td>17</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>15</td>
<td>113</td>
</tr>
<tr>
<td>dynamic-gamer</td>
<td>3</td>
<td>3</td>
<td>10</td>
<td>15</td>
<td>14</td>
<td>0</td>
<td>17</td>
<td>3</td>
<td>19</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>99</td>
</tr>
<tr>
<td>all-paca</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>6</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>12</td>
<td>98</td>
</tr>
<tr>
<td>cedalion</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>14</td>
<td>5</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>93</td>
</tr>
<tr>
<td>metis</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>91</td>
</tr>
<tr>
<td>nucelar</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>15</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>0</td>
<td>7</td>
<td>13</td>
<td>90</td>
</tr>
<tr>
<td>rlazya</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>17</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>88</td>
</tr>
<tr>
<td>gamer</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>18</td>
<td>13</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>5</td>
<td>83</td>
</tr>
<tr>
<td>hflow</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>53</td>
</tr>
<tr>
<td>miplan</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td>47</td>
</tr>
<tr>
<td>dpmplan</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td>43</td>
</tr>
<tr>
<td>hpp-ce</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>hpp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
</tr>
</tbody>
</table>
Outline

1. Introduction
   - Cost-Optimal Planning

2. Symbolic Search
   - (Background) Symbolic Search
   - Image Computation
   - State Invariants

3. Abstraction Heuristics
   - (Background) Abstractions
   - Merge-and-Shrink for Symbolic Search
   - Symbolic Perimeter Merge-and-Shrink

4. Symbolic Bidirectional Heuristic Search

5. Conclusions
   - Final Results: IPC14
   - Conclusions
Conclusions

Symbolic search for cost-optimal planning:
  - Analysis of image computation
  - State-invariant pruning

M&S heuristics in symbolic search planning

SPM&S: new perimeter abstraction heuristic based in symbolic search and M&S

Big question: can we use heuristics in symbolic planning?
  1. Used M&S and SPM&S in BDDA*
  2. SymBA*: symbolic bidirectional search + perimeter abstractions
Conclusions

- Symbolic bidirectional blind search
  - Currently, the best method for cost-optimal planning (only beaten by heuristics in domains where the heuristics are very precise).

- SPM&S: state-of-the-art heuristic

- Highlighted the relevance of symbolic search and regression

- Synergy of symbolic bidirectional search and perimeter abstractions
List of Publications

Álvaro Torralba, Stefan Edelkamp, and Peter Kissmann. Transition trees for cost-optimal symbolic planning. In *ICAPS*, 2013

Álvaro Torralba and Vidal Alcázar. Constrained symbolic search: On mutexes, BDD minimization and more. In *SoCS*, 2013


Thank you for your attention!

Questions?