Structural Patterns Heuristics via Fork Decomposition

Michael Katz    Carmel Domshlak

Technion, IE&M

ICAPS-08
Abstraction-based Admissible Heuristics for Cost-Optimal Classical Planning

Classical Planning

Planning task is 5-tuple $\langle V, A, C, s^0, G \rangle$:
- $V$: finite set of finite-domain state variables
- $A$: finite set of actions of form $\langle \text{pre}, \text{eff} \rangle$ (preconditions/effects; partial variable assignments)
- $C : A \mapsto \mathbb{R}^{0+}$ captures action cost
- $s^0$: initial state (variable assignment)
- $G$: goal description (partial variable assignment)
Abstraction-based **Admissible Heuristics for Cost-Optimal Classical Planning**

**Cost-Optimal Planning**

- **Given:** planning task $\Pi = \langle V, A, s^0, G \rangle$
- **Find:** operator sequence $a_1 \ldots a_n \in A^*$ transforming $s^0$ into some state $s_n \supseteq G$, while minimizing $\sum_{i=1}^{n} C(a_i)$
- **Approach:** $A^* +$ admissible heuristic $h : S \rightarrow \mathbb{R}^{0+}$
Abstraction-based Admissible Heuristics for Cost-Optimal Classical Planning

Abstraction heuristics

Heuristic estimate is goal distance in abstracted state space $S'$

Well-known: projection (pattern database) heuristics
Here we: both generalize and enhance them
Transition Graphs

Transition graph

**TG-structure** $\mathcal{T} = (S, L, Tr, s^0, S^*)$:

- $S$: finite set of states
- $L$: finite set of transition labels
- $Tr \subseteq S \times L \times S$: labelled transitions
- $s^0 \in S$: initial state
- $S^* \subseteq S$: goal states

Transition graph $\langle \mathcal{T}, \varpi \rangle$:

- $\mathcal{T}$: TG-structure with labels $L$
- transition cost function $\varpi : L \to \mathbb{R}^{0+}$

(Transition graph of planning task defined in the obvious way.)
Transition Graphs

Transition graph

TG-structure $\mathcal{T} = (S, L, \text{Tr}, s^0, S^*)$:

- $S$: finite set of states
- $L$: finite set of transition labels
- $\text{Tr} \subseteq S \times L \times S$: labelled transitions
- $s^0 \in S$: initial state
- $S^* \subseteq S$: goal states

Transition graph $\langle \mathcal{T}, \varpi \rangle$:

- $\mathcal{T}$: TG-structure with labels $L$
- transition cost function $\varpi : L \mapsto \mathbb{R}^{0+}$

(Transition graph of planning task defined in the obvious way.)
(Additive) Abstractions

Definition (additive abstractions)

Additive abstraction of transition graph $\langle T, \omega \rangle$ is

$$\left\{ \left\langle \left\langle T_i, \omega_i \right\rangle, \alpha_i \right\rangle \right\}_{i=1}^m$$

where

- $\langle T_i, \omega_i \rangle$: transition graph
- $\alpha_i$ maps states of $T$ to states of $T_i$ such that
  - initial state maps to initial state
  - goal states map to goal states
- holds $\sum_{i=1}^m d(\alpha_i(s), \alpha_i(s')) \leq d(s, s')$

Abstraction heuristic:

$$h(s) = \sum_{i=1}^m d(\alpha_i(s), S_i^*)$$

is (trivially) admissible
Widely-exploited idea: projections

map states to abstract states with perfect hash function

Definition (projection)

Projection $\Pi^{[V']} \quad \text{to variables } V' \subseteq V$: homomorphism $\alpha$ where

$\alpha(s) = \alpha(s') \iff s \text{ and } s' \text{ agree on } V'$

Each $a \in A$ satisfies $C(a) \geq \sum_{i=1}^{m} C_i(a[V_i])$
Problems of Projections

No tricks: abstract spaces are searched **exhaustively**

\[ \therefore \text{must keep number of reflected variables in each projection small} \leq O(\log(|V|)) \]

\[ \therefore \text{(often) price in heuristic accuracy in long-run} \]
Structural Abstraction Heuristics: Main Idea

Objective

(Katz & D, 2008a):

Instead of perfectly reflecting a few state variables, reflect many (up to $\Theta(|V|)$) state variables, BUT

♠ guarantee abstract space can be searched (implicitly) in poly-time
**Structural Abstraction Heuristics: Main Idea**

<table>
<thead>
<tr>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Katz &amp; D, 2008a):</td>
</tr>
<tr>
<td>Instead of perfectly reflecting a few state variables, reflect many (up to $\Theta(</td>
</tr>
<tr>
<td>♠ guarantee abstract space can be searched (implicitly) in poly-time</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstracting $\Pi$ by an instance of a <strong>tractable fragment</strong> of cost-optimal planning</td>
</tr>
<tr>
<td>☹ not many such known tractable fragments</td>
</tr>
<tr>
<td>☻ should find more, and useful for us!</td>
</tr>
</tbody>
</table>
Running Example
Adapted from Malte Helmert

\[ V = \{ p_1, p_2, c_1, c_2, c_3, t \} \]
\[
\text{dom}(p_1) = \text{dom}(p_2) = \{ A, B, C, D, E, F, G, c_1, c_2, c_3, t \}
\]
\[
\text{dom}(c_1) = \text{dom}(c_2) = \{ A, B, C, D \}
\]
\[
\text{dom}(c_3) = \{ E, F, G \}
\]
\[
\text{dom}(t) = \{ D, E \}
\]
\[
s^0, G \mapsto \text{see picture}
\]
\[
A \mapsto \text{loads, unloads, single-segment movements}
\]
Fork-Decomposition (Additive Abstractions)

$\Pi$

\[ \left\{ \Pi_{G_{v}^{f}}, \Pi_{G_{v}^{if}} \right\}_{v \in V} \]

$\Pi_{G_{c_{1}}^{f}}$

$CG(\Pi_{c_{1}}^{f})$

$CG(\Pi_{p_{1}}^{if})$

$\Pi_{G_{p_{1}}^{if}}$

$CG(\Pi_{p_{1}}^{if})$

+ ensuring proper action cost partitioning
Forks and Inverted Forks are Hard ...

- Even non-optimal planning for problems with fork and inverted fork causal graphs is NP-complete (Domshlak & Dinitz, 2001).

- Even if the domain-transition graphs of all variables are strongly connected, optimal planning for forks and inverted forks remains NP-hard (Helmert, 2003-04).

〜 Shall we give up?
Tractable Cases of Planning with Forks

**Theorem (forks)**

Cost-optimal planning for fork problems with root $r \in V$ is poly-time if

(i) $|\text{dom}(r)| = 2$, or
(ii) for all $v \in V$, we have $|\text{dom}(v)| = O(1)$,

**Theorem (inverted forks)**

Cost-optimal planning for (1-dependent) inverted fork problems with root $r \in V$ is poly-time if $|\text{dom}(r)| = O(1)$.
Mixing Causal-Graph & Variable-Domain Decompositions

\[ \Pi \]

\[ CG(\Pi) \]

\[ \{\Pi_G^{f_v}, \Pi_G^{if_v}\}_{v \in V} \]

\[ \Pi_G^{f_{c_1}} \]

\[ \Pi_G^{if_{p_1}} \]

\[ \phi_{c_1,i} : \text{dom}(c_1) \mapsto \{0, 1\} \]

\[ \phi'_{p_1,i} : \text{dom}(p_1) \mapsto \{0, \ldots, k\} \]

+ ensuring proper action cost partitioning
Back to our example

\[ \{ \Pi_{G_v}^f, \Pi_{G_v}^i \} \}_{v \in V} \]

\[ \phi_{c_1,i} : \text{dom}(c_1) \mapsto \{0, 1\} \]

\[ \phi'_{p_1,i} : \text{dom}(p_1) \mapsto \{0, 1, 2\} \]
Informative?

(Intractable) Fork Decomposition

\[ d(s^0, S_G) = 19 \quad h_{\text{max}} = 8 \quad h^2 = 13 \quad h_{\text{FF}} = 15 \]

- \( h_{\text{max}} \) (Bonet & Geffner, 2001)
- \( h^2 \) (Haslum & Geffner, 2000)
Informative?

(Intractable) Fork Decomposition

\[ d(s^0, S_G) = 19 \quad h_{\text{max}} = 8 \quad h^2 = 13 \quad h^{FF} = 15 \]

(Tractable) Fork + Variable-Domains Decomposition

\[ d(s^0, S_G) = 19 \quad h_{\text{max}} = 8 \quad h^2 = 13 \quad h^{FF} = 16 \]

Hmm ... what?

Further abstraction gives a more precise estimate??
Informative?

(Intractable) Fork Decomposition

\[ d(s^0, S_G) = 19 \quad h_{\text{max}} = 8 \quad h^2 = 13 \quad h^{\text{FI}} = 15 \]

(Tractable) Fork + Variable-Domains Decomposition

\[ d(s^0, S_G) = 19 \quad h_{\text{max}} = 8 \quad h^2 = 13 \quad h^{\text{FI}} = 16 \]

Hmm ... yes, that is possible!

Variable-domains abstraction may eliminate certain dependencies between the variables

\[ \leadsto \text{less dependencies} \leadsto \text{less action representatives} \leadsto \text{less action cost erosion} \leadsto \text{(potentially) higher estimate} \]
Performance Evaluation

Option 1: Empirical evaluation

Implement $h$, plug into A*, test (comparatively) on standard benchmark suites

- ☺ standard approach, per-problem-instance comparison
- ☹ no conclusions *a la*
  
  "$h$ expands fewer nodes than $h'$ on a benchmark suite $X"
Performance Evaluation

Option 1: Empirical evaluation
Implement $h$, plug into A*, test (comparatively) on standard benchmark suites

Option 2: Asymptotic performance analysis (Helmert and Mattmüller, 2008)
Given suite $\mathcal{D}$ and heuristic $h$, find a value $\alpha(h, \mathcal{D}) \in [0, 1]$ such that

(i) for all states $s$ in all problems $\Pi \in \mathcal{D}$,
   \[ h(s) \geq \alpha(h, \mathcal{D}) \cdot h^*(s) + o(h^*(s)) \]

(ii) there exist $\{\Pi_n\}_{n \in \mathbb{N}} \subseteq \mathcal{D}$ and solvable states $\{s_n\}_{n \in \mathbb{N}}$ with $s_n \in \Pi_n$, $\lim_{n \to \infty} h^*(s_n) = \infty$, and
   \[ h(s_n) \leq \alpha(h, \mathcal{D}) \cdot h^*(s_n) + o(h^*(s_n)) \]
Asymptotic Performance Ratios
Selected benchmark suites

<table>
<thead>
<tr>
<th>Domain</th>
<th>$h^+$</th>
<th>$h^k$</th>
<th>$h^{PDB}$</th>
<th>$h_{add}^{PDB}$</th>
<th>$h^g$</th>
<th>$h^j$</th>
<th>$h^{gj}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gripper</td>
<td>2/3</td>
<td>0</td>
<td>0</td>
<td>2/3</td>
<td>2/3</td>
<td>1/2</td>
<td>2/3</td>
</tr>
<tr>
<td>Logistics</td>
<td>3/4</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>Blocksworld</td>
<td>1/4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Miconic</td>
<td>6/7</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>5/6</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>Satellite</td>
<td>1/2</td>
<td>0</td>
<td>0</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
</tr>
</tbody>
</table>

ratios for $h^+$, $h^k$, $h^{PDB}$, $h_{add}^{PDB}$ are by Helmert and Mattmüller, 2008.
## Asymptotic Performance Ratios

### Selected benchmark suites

<table>
<thead>
<tr>
<th>Domain</th>
<th>$h^+$</th>
<th>$h^k$</th>
<th>$h_{PDB}^P$</th>
<th>$h_{add}^{PDB}$</th>
<th>$h^{\mathcal{F}}$</th>
<th>$h^{\mathcal{J}}$</th>
<th>$h^{\mathcal{FJ}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gripper</td>
<td>2/3</td>
<td>0</td>
<td>0</td>
<td>2/3</td>
<td>2/3</td>
<td>1/2</td>
<td>2/3</td>
</tr>
<tr>
<td>Logistics</td>
<td>3/4</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>Blocksworld</td>
<td>1/4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Miconic</td>
<td>6/7</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>5/6</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>Satellite</td>
<td>1/2</td>
<td>0</td>
<td>0</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
</tr>
</tbody>
</table>

$h^\text{PDB}_{\text{add}}$: optimal, manually-selected set of projections

$h^{\mathcal{FJ}}$: non-parametric set of abstractions

Basic variable-domain abstractions to binary/ternary
## Summary

### What we do

From small projections to large structural abstractions

### Future work

- more tractability results for cost-optimal planning!
- optimization of variable-domains abstraction
- approximation-oriented structural patterns
- ...
- implementation and empirical evaluation