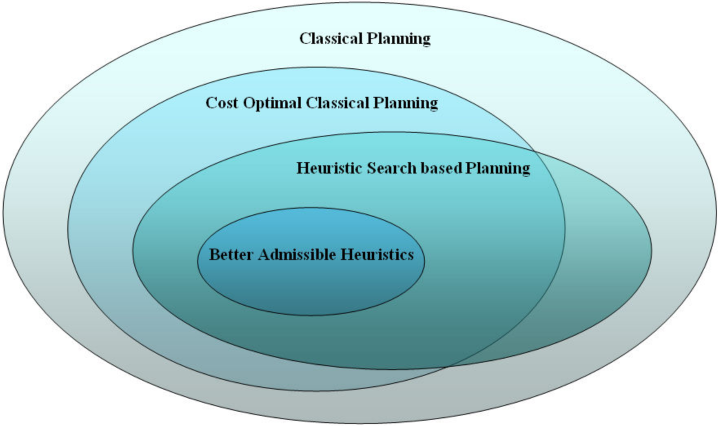
Pushing the Envelope of Abstraction-based Admissible Heuristics

Context


Classical Planning
Cost Optimal Classical Planning
Heuristic Search based Planning
Better Admissible Heuristics

Explaining the Context

New Abstraction-based Admissible Heuristics for Cost-Optimal Classical Planning

Explaining the Context

New Abstraction-based Admissible Heuristics for Cost-Optimal Classical Planning

Classical Planning

Planning task is 5-tuple $\langle V, A, \mathcal{C}, s^0, G \rangle$:

- 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)
- $\mathcal{C} : A \mapsto \mathbb{R}^{0+}$ captures action cost
- s^0: initial state (variable assignment)
- G: goal description (partial variable assignment)

Explaining the Context

New 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 $\sum_{i=1}^{n} \mathcal{C}(a_i)$

Approach: A* + admissible heuristic $h : S \mapsto \mathbb{R}^{0+}$

Explaining the Context

New 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 $a_1 \dots a_n \in A^*$
transforming s^0 into some state $s_n \supseteq G$,
while minimizing $\sum_{i=1}^{n} \mathcal{C}(a_i)$

Approach: A* $+$ admissible heuristic $h : S \mapsto \mathbb{R}^{0+}$

Admissible $\equiv$ underestimate goal distance

Explaining the Context

New 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

Explaining the Context

New 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 $\mathcal{T} = (S, L, Tr, s^0, S^\star)$:

- 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
- $S^\star \subseteq S$: 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.)

Transition Graphs

Transition graph

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

- 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
- $S^\star \subseteq S$: 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 $\langle \mathcal{T}, \varpi \rangle$ is $\{\langle\langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i \rangle\}_{i=1}^{m}$ where

- $\langle \mathcal{T}_i, \varpi_i \rangle$: transition graph
- α_i maps states of $\mathcal{T}$ to states of $\mathcal{T}_i$ 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:
$h(s) = \sum_{i=1}^{m} d(\alpha_i(s), S_i^\star)$ is (trivially) admissible

(Additive) Abstractions

Definition (additive abstractions)

Additive abstraction of transition graph $\langle \mathcal{T}, \varpi \rangle$ is $\{\langle\langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i\rangle\}_{i=1}^{m}$ where

- $\langle \mathcal{T}_i, \varpi_i \rangle$: transition graph
- α_i maps states of $\mathcal{T}$ to states of $\mathcal{T}_i$ 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:

$h(s) = \sum_{i=1}^{m} d(\alpha_i(s), S_i^{\star})$ is (trivially) admissible

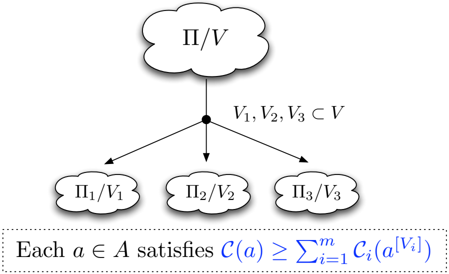
Projections

Widely-exploited idea: projections

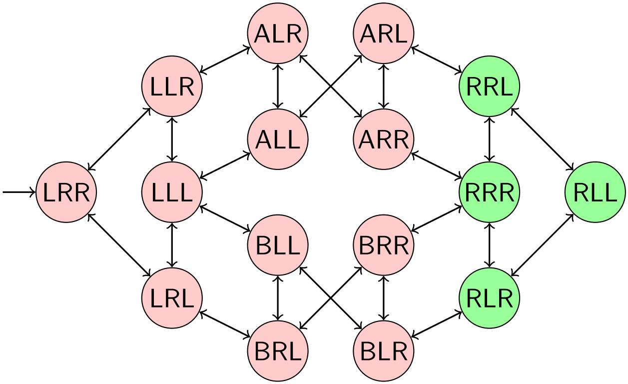
$\leadsto$ map states to abstract states with perfect hash function

Definition (projection)

Projection $\Pi^{[V']}$ to variables $V' \subseteq V$: homomorphism α where $\alpha(s) = \alpha(s')$ iff s and s' agree on V'



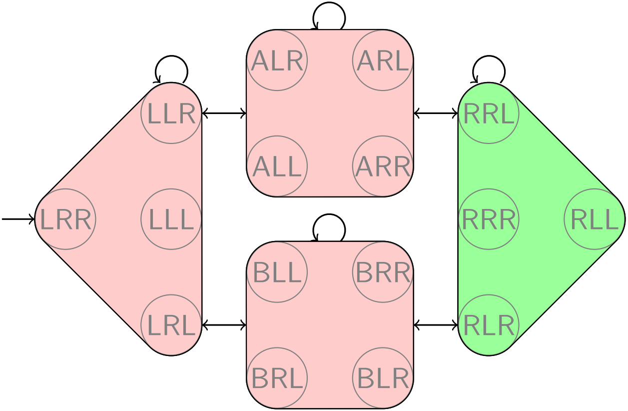
Example
Copyrights: Malte Helmert



- one package, two trucks, two locations
- state variable package: $\{L, R, A, B\}$
- state variable truck A: $\{L, R\}$
- state variable truck B: $\{L, R\}$

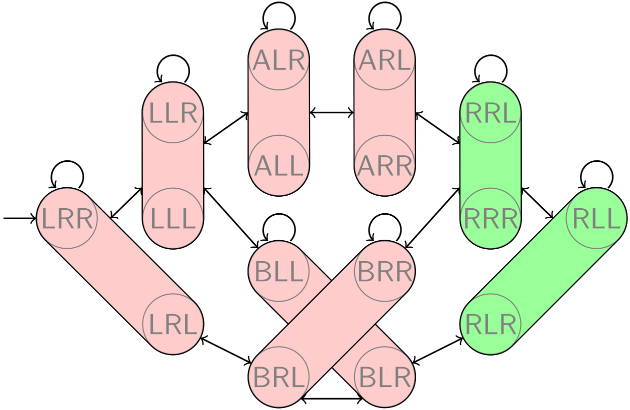
Example: Projection (1)

Project to {package}:



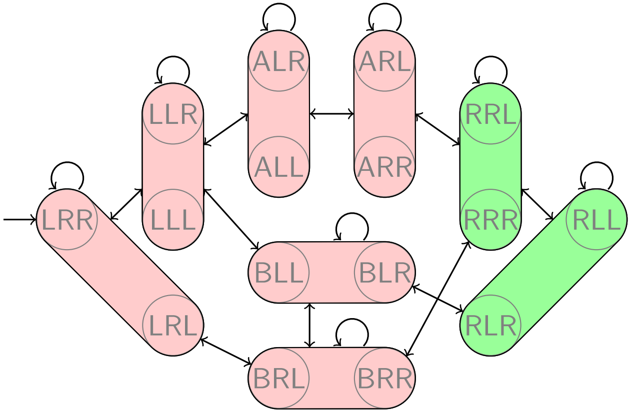
Example: Projection (2)

Project to {package, truck A}:



Example: Projection (2)

Project to {package, truck A}:

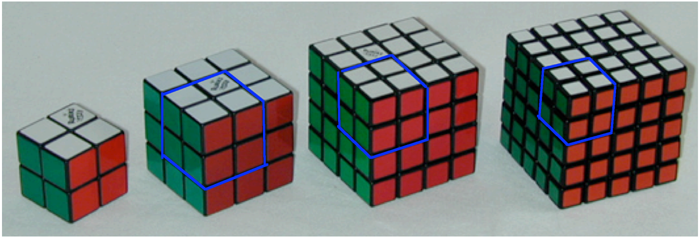


Problems of Projections

No tricks: abstract spaces are searched exhaustively

$\leadsto$ must keep number of reflected variables in each projection small ($\leq O(\log(|V|))$)

$\leadsto$ (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

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

How

Abstracting Π by an instance of a tractable fragment of cost-optimal planning

- ☹ not many such known tractable fragments
- ☺ should find more, and useful for us!

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

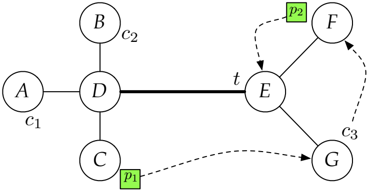
How

Abstracting Π by an instance of a tractable fragment of cost-optimal planning

- ☹ not many such known tractable fragments
- ☺ should find more, and useful for us!

Here Come the Forks!

Running Example

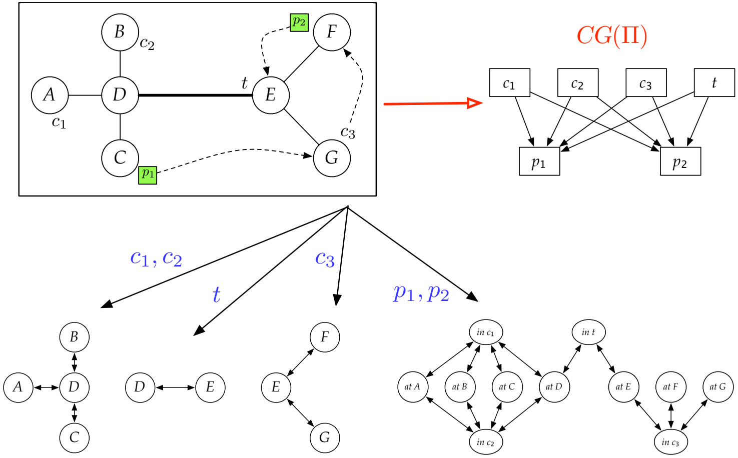


$$
\begin{aligned}
V &= \{p_1, p_2, c_1, c_2, c_3, t\} \\
dom(p_1) &= dom(p_2) = \{A, B, C, D, E, F, G, c_1, c_2, c_3, t\} \\
dom(c_1) &= dom(c_2) = \{A, B, C, D\} \\
dom(c_3) &= \{E, F, G\} \\
dom(t) &= \{D, E\}
\end{aligned}
$$

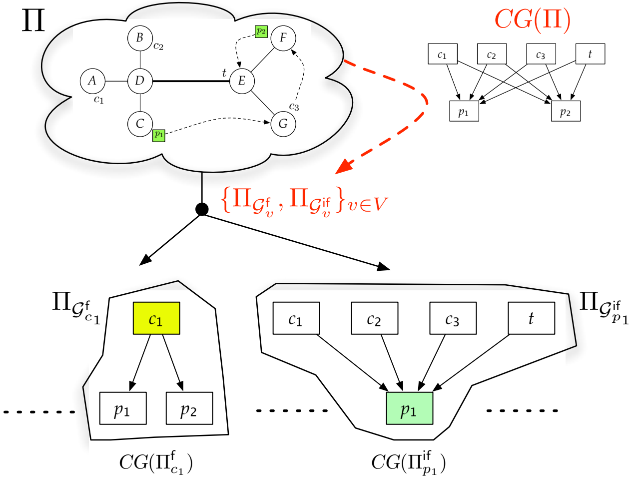
$s^0, G \mapsto$ see picture

$A \mapsto$ loads, unloads, single-segment movements

Causal Graph + Domain Transition Graphs

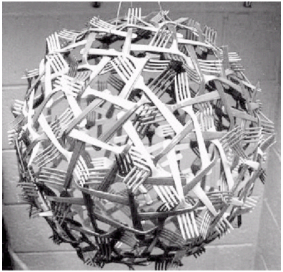


Fork-Decomposition (Additive Abstractions)



+ ensuring proper action cost partitioning

Action Cost Partitioning = Gluing Things Together



Works?
Problem!

Forks and Inverted Forks are Hard ...

- ☹ Even non-optimal planning for problems with fork and inverted fork causal graphs is NP-complete (D & 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).

$\leadsto$ 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

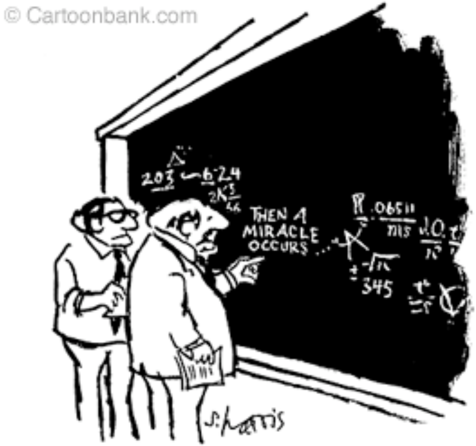
(i) $|dom(r)| = 2$, *or*

(ii) *for all $v \in V$, we have $|dom(v)| = O(1)$,*

Theorem (inverted forks)

Cost-optimal planning for inverted fork problems with root $r \in V$ is poly-time if $|dom(r)| = O(1)$.

Tractable Cases of Planning with Forks



"I think you should be more explicit here in step two."

Theorem (inverted forks)

Theorem (inverted forks)

Cost-optimal planning for inverted fork problems with root $r \in V$ is poly-time if $|dom(r)| = d = O(1)$.

Proof sketch (Construction)

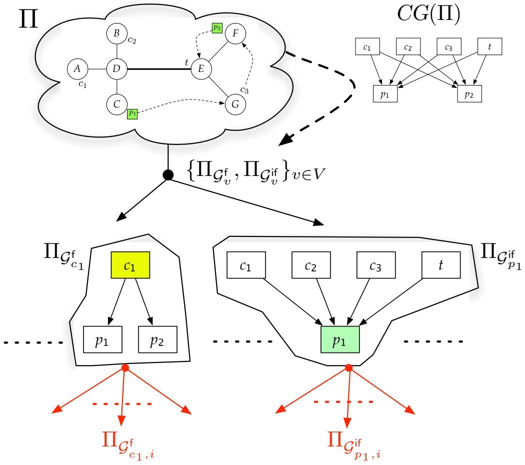
(1) Create all $\Theta(d^d)$ cycle-free paths from $s^0[r]$ to $G[r]$ in *DTG*(r, Π).

(2) For each $u \in \text{pred}(r)$, and each $x, y \in dom(u)$, compute the cost-minimal path from x to y in *DTG*(u, Π).

(3) For each path in *DTG*(r, Π) generated in step (1), construct a plan for Π based on that path for r, and the shortest paths computed in (2).

(4) Take minimal cost plan from (3).

Mixing Causal-Graph & Variable-Domain Decompositions

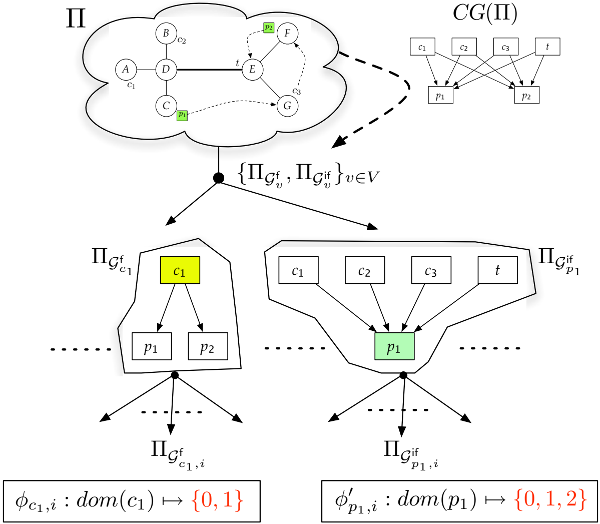


$\phi_{c_1,i} : dom(c_1) \mapsto \{0,1\}$

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

$+$ ensuring proper action cost partitioning

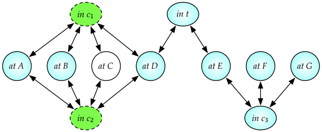
Back to our example



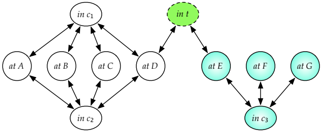
Back to our example

$$\forall l \in Dom(p_1): \quad \phi'_{p_1,i}(l) = \begin{cases} 0, d(I[p_1], l) < 2i-1 \\ 1, d(I[p_1], l) = 2i-1 \\ 2, d(I[p_1], l) > 2i-1 \end{cases}$$

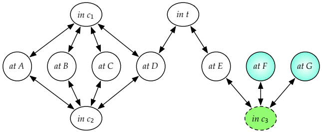
$\phi'_{p_1,1}$



$\phi'_{p_1,2}$



$\phi'_{p_1,3}$



Informative?

(Intractable) Fork Decomposition

$d(s^0, S_G) = 19$ $h_{\max} = 8$ $h^2 = 13$ $h^{\mathfrak{FI}} = 15$

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

Informative?

(Intractable) Fork Decomposition

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

(Tractable) Fork + Variable-Domains Decomposition

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

Hmm ... what?

Further abstraction gives a more precise estimate??

Informative?

(Intractable) Fork Decomposition

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

(Tractable) Fork + Variable-Domains Decomposition

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

Hmm ... yes, that is possible!

Variable-domains abstraction may eliminate certain dependencies between the variables
$\leadsto$ less dependencies $\leadsto$ less action representatives $\leadsto$ less action cost erosion $\leadsto$ (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

Domain	h^+	h^k	h^{PDB}	$h^{\mathsf{PDB}}_{\mathsf{add}}$	$h^{\mathcal{F}}$	$h^{\mathcal{I}}$	$h^{\mathcal{FI}}$
GRIPPER	2/3	0	0	2/3	2/3	1/2	2/3
LOGISTICS	3/4	0	0	1/2	1/2	1/2	1/2
BLOCKSWORLD	1/4	0	0	0	0	0	0
MICONIC	6/7	0	0	1/2	5/6	1/2	1/2
SATELLITE	1/2	0	0	1/6	1/6	1/6	1/6

ratios for h^+, h^k, h^{PDB}, $h^{\mathsf{PDB}}_{\mathsf{add}}$ are by Helmert and Mattmüller, 2008.

Asymptotic Performance Ratios

Selected benchmark suites

Domain	h^+	h^k	h^{PDB}	$h^{\mathsf{PDB}}_{\mathsf{add}}$	$h^{\mathfrak{F}}$	$h^{\mathfrak{I}}$	$h^{\mathfrak{FI}}$
Gripper	$2/3$	0	0	$2/3$	$2/3$	$1/2$	$2/3$
Logistics	$3/4$	0	0	$1/2$	$1/2$	$1/2$	$1/2$
Blocksworld	$1/4$	0	0	0	0	0	0
Miconic	$6/7$	0	0	$1/2$	$5/6$	$1/2$	$1/2$
Satellite	$1/2$	0	0	$1/6$	$1/6$	$1/6$	$1/6$

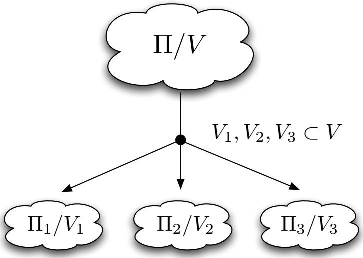
$h^{\mathsf{PDB}}_{\mathsf{add}}$: optimal, manually-selected set of projections

$h^{\mathfrak{FI}}$: non-parametric set of abstractions
basic variable-domain abstractions to binary/ternary

Action-Cost Partitioning: Back to Projections

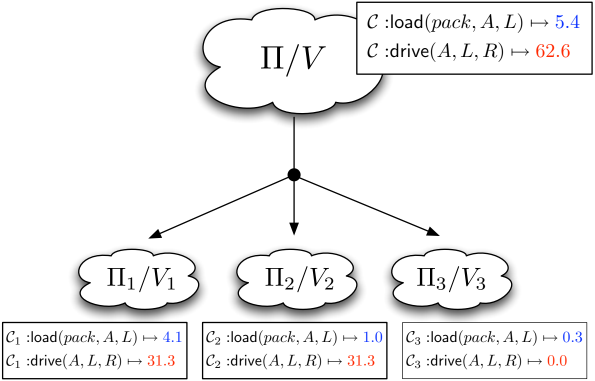
Definition (projection)

Projection $\Pi^{[V']}$ to variables $V' \subseteq V$: homomorphism α where $\alpha(s) = \alpha(s')$ iff s and s' agree on V'

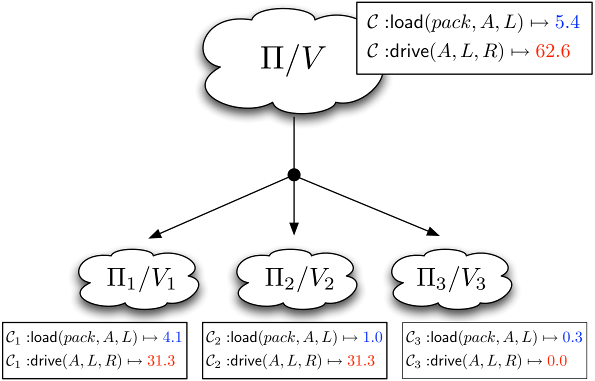


Each $a \in A$ satisfies $\mathcal{C}(a) \geq \sum_{i=1}^{m} \mathcal{C}_i(a^{[V_i]})$

Action-Cost Partitioning: Back to Projections



Action-Cost Partitioning: Back to Projections



need selecting a good action-cost partition

$\rightsquigarrow$ optimal action-cost partition?

Optimizing Action-Cost Partitioning

Pitfalls

- ☹ infinite space of choices
- ☹ decision process should be fully unsupervised
- ☹ decision process should be state-dependent

⇝ *"determining which abstractions [action-cost partitions] will produce additives that are better than max over standards is still a big research issue."* (Yang *et al.*, JAIR, 2008)

Main Idea

Main Idea

(Katz & D, 2008b):

Instead of searching each abstract transition graph $\langle \mathcal{T}_i, \varpi_i \rangle$ given an action cost partition using dynamic programming

1. compile SSSP problem over each TG-structure $\mathcal{T}_i$ into a linear program $\mathscr{L}_i$ with action-costs being free variables
2. combine $\mathscr{L}_1, \dots, \mathscr{L}_m$ with additivity constraints $\mathcal{C}(a) \geq \sum_{i=1}^{m} \mathcal{C}_i(a^{[V_i]})$
3. solution of the joint LP $\rightsquigarrow$ $\rightsquigarrow h(s)$ under optimal action-cost partition

Main Idea

Main Idea

(Katz & D, 2008b):

Instead of searching each abstract transition graph $\langle \mathcal{T}_i, \varpi_i \rangle$ given an action cost partition using dynamic programming

1. compile SSSP problem over each TG-structure $\mathcal{T}_i$ into a linear program $\mathscr{L}_i$ with action-costs being free variables
2. combine $\mathscr{L}_1, \ldots, \mathscr{L}_m$ with additivity constraints $\mathcal{C}(a) \geq \sum_{i=1}^{m} \mathcal{C}_i(a^{[V_i]})$
3. solution of the joint LP $\rightsquigarrow$
   $\rightsquigarrow h(s)$ under optimal action-cost partition

Single-Source Shortest Paths: LP Formulation

LP formulation

Given: digraph $G = (N, E)$, source node $v \in N$

LP variables: $d(v') \rightsquigarrow$ shortest-path length from v to v'

LP:

$$\max_{\vec{d}} \sum_{v'} d(v')$$

$$\text{s.t. } d(v) = 0$$

$$d(v'') \leq d(v') + w(v', v''), \quad \forall (v', v'') \in E$$

Step 1: Compile SSSP over $\mathcal{T}_i$ into $\mathscr{L}_i$

LP formulation

Given: TG-structure $\mathcal{T}_i$, state s

LP variables: $\{d(s') \mid s' \in S_i\} \cup \{d(S_i^\star)\} \cup \{w(a,i)\}$

LP:

$$\max \quad d(S_i^\star)$$

$$\text{s.t.} \quad \begin{cases} d(s') \leq d(s'') + w(a,i), & \forall \langle s'', a, s' \rangle \in Tr_i \\ d(s') = 0, & s' = s^{[V_i]} \\ d(S_i^\star) \leq d(s'), & s' \in S_i^\star \end{cases}$$

Step 2: Properly combine $\{\mathscr{L}_i\}_{i=1}^m$

LP formulation

Given: TG-structure $\{\mathcal{T}_i\}_{i=1}^m$ state s

LP variables: $\bigcup_{i=1}^m \{d(s') \mid s' \in S_i\} \cup \{d(S_i^\star)\} \cup \{w(a,i)\}$

LP:

$$\max \quad \sum_{i=1}^{m} d(S_i^\star)$$

$$\text{s.t.} \quad \forall i \quad \begin{cases} d(s') \le d(s'') + w(a,i), & \forall \langle s'', a, s' \rangle \in Tr_i \\ d(s') = 0, & s' = s^{[V_i]} \\ d(S_i^\star) \le d(s'), & s' \in S_i^\star \end{cases}$$

$$\forall a \in A : \quad \sum_{i=1}^{m} w(a,i) \le \mathcal{C}(a)$$

Optimizing Action-Cost Partitioning: Generalization

General theory of LP-optimizable ensembles
of additive heuristic functions

- Warning: Any reduction to LP is not enough
  $\rightsquigarrow$ requires (surprising) relation between polyhedron and planning problem

Optimizing Action-Cost Partitioning: Generalization

General theory of LP-optimizable ensembles
of additive heuristic functions

- Warning: Any reduction to LP is not enough
- Works as above for
  - projection and variable-domain abstraction (PDB) heuristics
  - constrained PDBs heuristics (Haslum *et al.*, 2005)
  - merge-and-shrink abstractions (Helmert *et al.*, 2007)
- Suitable poly-size LPs $\mathscr{L}_i$ exist also for
  - fork-decomposition heuristics
  - tree-COP reducible fragments of tractable cost-optimal planning (from Katz & D, 2007)
  - ...

Optimizing Action-Cost Partitioning: Generalization

General theory of LP-optimizable ensembles of additive heuristic functions

- Warning: Any reduction to LP is not enough
- Works as above for
  - projection and variable-domain abstraction (PDB) heuristics
  - constrained PDBs heuristics (Haslum *et al.*, 2005)
  - merge-and-shrink abstractions (Helmert *et al.*, 2007)
- Suitable poly-size LPs $\mathscr{L}_i$ exist also for
  - fork-decomposition heuristics
  - tree-COP reducible fragments of tractable cost-optimal planning (from Katz & D, 2007)
  - ...

LP for Inverted Forks (1)

Given: problem Π, state s, goal G

Variables

$$\overrightarrow{x} = \{h^*\} \cup \bigcup_{\substack{v \in V' \setminus \{r\}, \\ \vartheta, \vartheta' \in dom(v)}} \{d(v, \vartheta, \vartheta')\}.$$

$d(v, \vartheta, \vartheta') \rightsquigarrow$ cost of the cheapest sequence of actions *affecting* v that changes its value from ϑ to ϑ'

Objective

$$\max \ \{h^*\}$$

LP for Inverted Forks (2)

Given: problem Π, state s, goal G

Constraints (I)

For each simple path $\langle a_1 \cdot \ldots \cdot a_m \rangle$ from $s[r]$ to $G[r]$ in $\mathit{DTG}(r, \Pi)$,

$$h^* \leq \sum_{v \in V \setminus \{r\}} d(v, s_0[v], s_1[v]) + \sum_{i=1}^{m} \left(\mathcal{C}(a_i) + \sum_{v \in V' \setminus \{r\}} d(v, s_i[v], s_{i+1}[v]) \right)$$

where

$$s_i[v] = \begin{cases} s[v], & i = 0 \\ G[v], & i = m+1, \text{ and } G[v] \text{ is specified} \\ \mathsf{pre}(a_i)[v], & 1 \leq i \leq m, \text{ and } \mathsf{pre}(a_i)[v] \text{ is specified} \\ s_{i-1}[v], & \text{otherwise} \end{cases}$$

Semantics: *The cost of solving the problem is not greater than the cost of any cycle-free path of r plus sums of costs of reaching the prevail conditions of actions on this path and reaching the goal afterwards.*

LP for Inverted Forks (3)

Given: problem Π, state s, goal G

Constraints (II)

For each $v \in V \setminus \{r\}$, $\vartheta \in dom(v)$,

$$d(v, \vartheta, \vartheta) = 0$$

For each v-changing action $a \in A$,

$$d(v, \vartheta, \mathsf{post}(a)[v]) \leq d(v, \vartheta, \mathsf{pre}(a)[v]) + \mathcal{C}(a)$$

Semantics: *Shortest-path constraints.*

Empirical Evaluation

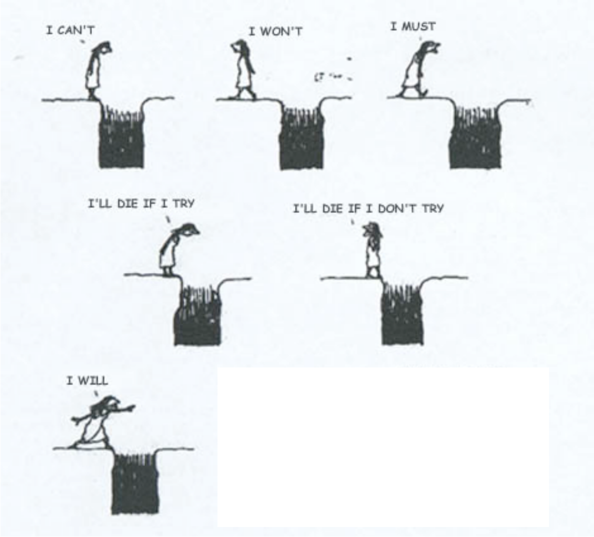
Are you crazy??

"Now, this is just a simulation of what the blocks will look like once they're assembled."

Empirical Evaluation

Are you crazy?? Well, depends on the moon's position ;)

Since September 2008 ...



Initial State Estimate / Logistics-00

Uniform action-cost partition

#	h^*	HHH_{10^5}	$h^{\mathcal{F}}$	$h^{\mathcal{J}}$	$h^{\mathcal{FJ}}$
01	20	20	20/20	18/20	18/20
02	19	19	19/19	15/19	16/19
03	15	15	15/15	11/14	12/15
04	27	27	27/27	24/26	24/27
05	17	17	17/17	14/17	14/17
06	8	8	8/8	7/7	7/8
07	25	25	25/25	21/24	22/25
08	14	14	14/14	11/14	12/14
09	25	25	25/25	22/25	22/25
10	36	36	36/36	30/35	30/36
11	44	42	43/43	36/43	36/44
12	31	31	31/31	26/28	26/31
13	44	43	44/44	38/43	38/44
14	36	35	36/36	30/34	30/36
15	30	30	30/30	26/28	26/30
16	45	27	45/45	36/44	36/45
17	42	36	42/42	34/39	34/42
18	48	39	48/48	40/45	40/48
19	60	54	59/59	50/57	50/60
20	42	36	42/42	33/38	34/42
21	68	43	67/67	58/66	58/68

Initial State Estimate / Logistics-00

From uniform to optimal action-cost partition

#	h^*	HHH_{10^5}	$h^{\mathcal{F}}$	$h^{\mathcal{I}}$	$h^{\mathcal{FI}}$
01	20	20	20/20	18/20	18/20
02	19	19	19/19	15/19	16/19
03	15	15	15/15	11/14	12/15
04	27	27	27/27	24/26	24/27
05	17	17	17/17	14/17	14/17
06	8	8	8/8	7/7	7/8
07	25	25	25/25	21/24	22/25
08	14	14	14/14	11/14	12/14
09	25	25	25/25	22/25	22/25
10	36	36	36/36	30/35	30/36
11	44	42	43/43	36/43	36/44
12	31	31	31/31	26/28	26/31
13	44	43	44/44	38/43	38/44
14	36	35	36/36	30/34	30/36
15	30	30	30/30	26/28	26/30
16	45	27	45/ 45	36/44	36/45
17	42	36	42/42	34/39	34/42
18	48	39	48/48	40/45	40/48
19	60	54	59/59	50/57	50/60
20	42	36	42/42	33/38	34/42
21	68	43	67/67	58/66	58/68

Planning / Logistics-00

Expanded nodes

#	h^*	HHH_{10^5}		$h^{\mathcal{F}}$		$h^{\mathcal{FJ}}$ + opt	
		nodes	*time*	*nodes*	*time*	*nodes*	*time*
01	20	21	0.05	21	10.49	21	20.82
02	19	20	0.04	20	10.4	20	20.36
03	15	16	0.05	16	5.18	16	10.85
04	27	28	0.33	28	22.81	28	47.42
05	17	18	0.34	18	11.72	18	21.63
06	8	9	0.33	9	2.99	9	8.89
07	25	26	1.11	26	26.88	26	53.81
08	14	15	1.12	15	10.37	15	21.19
09	25	26	1.14	26	27.78	26	51.52
10	36	37	4.55	37	426.07	37	973.46
11	44	2460	4.65	1689	14259.8	45	1355.23
12	31	32	6.5	32	374.48	32	876.9
13	44	7514	6.84	45	702.29	45	1621.74
14	36	37	8.94	37	474.8	37	1153.85
15	30	31	8.84	31	448.86	31	1052.46
16	45	29319	17.35	46	3517.25	46	7635.96
17	42	1561610	45.61	43	3297.69	43	7192.51
18	48	199428	24.95			49	10014.3
19	60					61	15625.5
20	42	6095	24.9	43	4325.45	43	9470.85
21	68					69	22928.4

Planning / Logistics-00
Expanded nodes and Time

#	h^*	HHH_{10^5}		$h^{\mathfrak{F}}$		$h^{\mathfrak{FI}}$ + opt	
		nodes	*time*	*nodes*	*time*	*nodes*	*time*
01	20	21	*0.05*	21	*10.49*	21	*20.82*
02	19	20	*0.04*	20	*10.4*	20	*20.36*
03	15	16	*0.05*	16	*5.18*	16	*10.85*
04	27	28	*0.33*	28	*22.81*	28	*47.42*
05	17	18	*0.34*	18	*11.72*	18	*21.63*
06	8	9	*0.33*	9	*2.99*	9	*8.89*
07	25	26	*1.11*	26	*26.88*	26	*53.81*
08	14	15	*1.12*	15	*10.37*	15	*21.19*
09	25	26	*1.14*	26	*27.78*	26	*51.52*
10	36	37	*4.55*	37	*426.07*	37	*973.46*
11	44	2460	*4.65*	1689	*14259.8*	45	*1355.23*
12	31	32	*6.5*	32	*374.48*	32	*876.9*
13	44	7514	*6.84*	45	*702.29*	45	*1621.74*
14	36	37	*8.94*	37	*474.8*	37	*1153.85*
15	30	31	*8.84*	31	*448.86*	31	*1052.46*
16	45	29319	*17.35*	46	*3517.25*	46	*7635.96*
17	42	1561610	*45.61*	43	*3297.69*	43	*7192.51*
18	48	199428	*24.95*			49	*10014.3*
19	60					61	*15625.5*
20	42	6095	*24.9*	43	*4325.45*	43	*9470.85*
21	68					69	*22928.4*

Planning / Logistics-00

Shall we redefine the notion of success?...

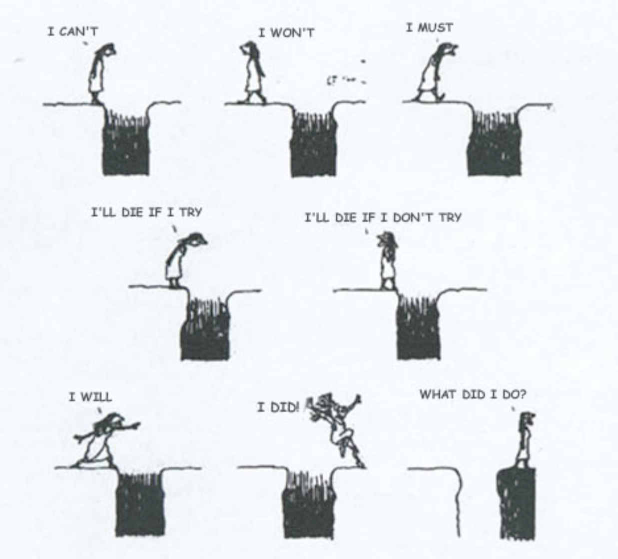
#	h^*	HHH_{10^5}		$h^{\mathcal{F}}$			$h^{\mathcal{FI}}$ + opt	
		nodes	*time*	*nodes*	*time*	♠	*nodes*	*time*
01	20	21	*0.05*	21	*10.49*		21	*20.82*
02	19	20	*0.04*	20	*10.4*		20	*20.36*
03	15	16	*0.05*	16	*5.18*		16	*10.85*
04	27	28	*0.33*	28	*22.81*		28	*47.42*
05	17	18	*0.34*	18	*11.72*		18	*21.63*
06	8	9	*0.33*	9	*2.99*		9	*8.89*
07	25	26	*1.11*	26	*26.88*		26	*53.81*
08	14	15	*1.12*	15	*10.37*		15	*21.19*
09	25	26	*1.14*	26	*27.78*		26	*51.52*
10	36	37	*4.55*	37	*426.07*		37	*973.46*
11	44	2460	*4.65*	1689	*14259.8*		45	*1355.23*
12	31	32	*6.5*	32	*374.48*		32	*876.9*
13	44	7514	*6.84*	45	*702.29*		45	*1621.74*
14	36	37	*8.94*	37	*474.8*		37	*1153.85*
15	30	31	*8.84*	31	*448.86*		31	*1052.46*
16	45	29319	*17.35*	46	*3517.25*		46	*7635.96*
17	42	1561610	*45.61*	43	*3297.69*		43	*7192.51*
18	48	199428	*24.95*				49	*10014.3*
19	60						61	*15625.5*
20	42	6095	*24.9*	43	*4325.45*		43	*9470.85*
21	68						69	*22928.4*

Planning / Logistics-00

No. Structural pattern databases!

#	h^*	HHH_{10^5}		$h^{\mathcal{F}}$			$h^{\mathcal{FI}}$ + opt	
		nodes	*time*	*nodes*	*time*	♠	*nodes*	*time*
01	20	21	*0.05*	21	*10.49*	*0.27*	21	*20.82*
02	19	20	*0.04*	20	*10.4*	*0.27*	20	*20.36*
03	15	16	*0.05*	16	*5.18*	*0.27*	16	*10.85*
04	27	28	*0.33*	28	*22.81*	*0.33*	28	*47.42*
05	17	18	*0.34*	18	*11.72*	*0.33*	18	*21.63*
06	8	9	*0.33*	9	*2.99*	*0.33*	9	*8.89*
07	25	26	*1.11*	26	*26.88*	*0.41*	26	*53.81*
08	14	15	*1.12*	15	*10.37*	*0.43*	15	*21.19*
09	25	26	*1.14*	26	*27.78*	*0.41*	26	*51.52*
10	36	37	*4.55*	37	*426.07*	*3.96*	37	*973.46*
11	44	2460	*4.65*	1689	*14259.8*	*4.25*	45	*1355.23*
12	31	32	*6.5*	32	*374.48*	*4.68*	32	*876.9*
13	44	7514	*6.84*	45	*702.29*	*4.63*	45	*1621.74*
14	36	37	*8.94*	37	*474.8*	*5.12*	37	*1153.85*
15	30	31	*8.84*	31	*448.86*	*5.12*	31	*1052.46*
16	45	29319	*17.35*	46	*3517.25*	*24.73*	46	*7635.96*
17	42	1561610	*45.61*	43	*3297.69*	*24.13*	43	*7192.51*
18	48	199428	*24.95*	697		*24.73*	49	*10014.3*
19	60			21959		*33.61*	61	*15625.5*
20	42	6095	*24.9*	43	*4325.45*	*29.61*	43	*9470.85*
21	68			106534		*61.54*	69	*22928.4*

Empirical Evaluation



Summary

Formal results on abstraction-based admissible heuristics

- from small projections to structural abstractions
- optimal combination of multiple abstractions

Ongoing and future work:

- structural pattern databases! *(in theaters in 2009?)*
- more tractability results for (cost-optimal) planning

- optimization of patterns selection
- optimization of variable-domains abstraction
- approximation-oriented structural patterns
- ...