Implicit Abstraction Heuristics for Cost-Optimal Planning

Michael Katz
Advisor: *Carmel Domshlak*

Technion - Israel Institute of Technology

Planning Language

Classical Planning in SAS$^+$

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)

How to Solve

Cost-Optimal Planning

Given: planning task $\Pi = \langle V, A, \mathcal{C}, s^0, G\rangle$

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 $h : S \mapsto \mathbb{R}^{0+}$

Admissible $\equiv$ underestimate goal distance

Abstractions

Abstraction

Abstraction is a pair of state space S' and mapping $\alpha : S \mapsto S'$ such that the goal distance is not increased

Abstractions

Abstraction

Abstraction is a pair of state space S' and mapping $\alpha : S \mapsto S'$ such that the goal distance is not increased

Abstraction heuristic

Heuristic estimate is goal distance in abstracted state space S'

Abstractions

Abstraction

Abstraction is a pair of state space S' and mapping $\alpha : S \mapsto S'$ such that the goal distance is not increased

Abstraction heuristic

Heuristic estimate is goal distance in abstracted state space S'

Well-known: explicit abstraction heuristics
Examples: projection (pattern database) heuristics
Merge and Shrink heuristics
Problem: abstract spaces are searched exhaustively
$\leadsto$ predefined constant bound on abstract space size

Contributions

1. Discovering new islands of tractability for both satisficing and cost-optimal planning.
2. Implicit abstraction heuristics for cost-optimal planning.
3. Optimal composition of abstraction heuristics.

Contributions

1. Discovering new islands of tractability for both satisficing and cost-optimal planning.
2. Implicit abstraction heuristics for cost-optimal planning.
3. Optimal composition of abstraction heuristics.

Planning is Hard!

Bad News

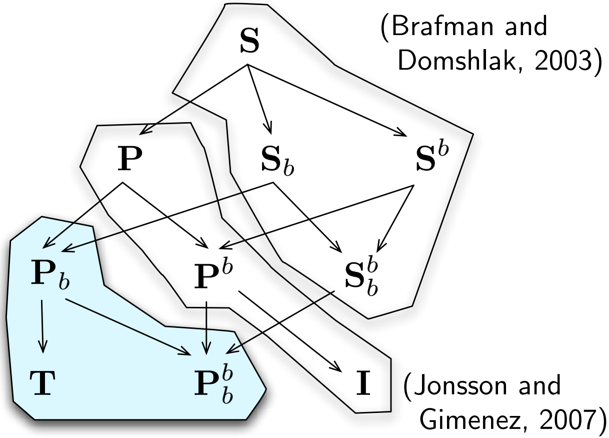
- planning is intractable in general (Chapman, 1987)
- even the "simple" classical planning with propositional state variables is PSPACE-complete (Bylander, 1994)

Worse News

- no difference in the theoretical complexity of satisficing and cost-optimal planning in the general case (Bylander, 1994)
- for a given domain cost-optimal planning is usually harder (Helmert, 2003)

Satisficing Planning Complexity – UB

S - Single connected
P - Polytree
T - Tree
I - Inverted Tree



subscript/superscript b refers to constant bound on in-degree/out-degree

Planning Complexity - Detailed Results for UB

ICAPS 2007, JAIR 2008

Cost-Optimal

	$k=1$	$k=2$	$k=3$	$k>3$	$k=\Theta(n)$
$\mathbf{P}_b$	—	—	—	—	P
$\mathbf{P}(k)$	P				NPC
$\mathbf{S}_b^b$	NPC	—	—	—	NPC

Satisficing

	$k=1$	$k=2$	$k=3$	$k>3$	$k=\Theta(n)$
$\mathbf{P}_b$	—	—	—	—	P
$\mathbf{P}(k)$	P	P	P		NPC
$\mathbf{S}_b^b$		NPC	—	—	NPC

k refers to k-dependence

Relaxing Domain Bounds

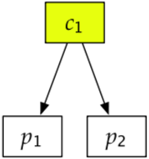
Tractable Cases of Planning with Forks, ICAPS 2008a

Theorem (forks)

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

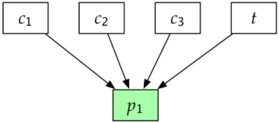
(i) $|dom(r)| = 2$, or

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



Theorem (inverted forks)

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



Contributions

1. Discovering new islands of tractability for both satisficing and cost-optimal planning.
2. Implicit abstraction heuristics for cost-optimal planning.
3. Optimal composition of abstraction heuristics.

Implicit Abstraction Heuristics: Basic Idea

Objective

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

Implicit Abstraction Heuristics: Basic Idea

Objective

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

Contribution

ICAPS 2008a, ICAPS 2009, JAIR 2010

1. acyclic causal-graph decompositions – a general framework for additive implicit abstractions that is based on decomposing the task at hand along its causal graph
2. fork decompositions – a concrete family of additive implicit abstractions, that are based on two novel fragments of tractable cost-optimal planning
3. databased implicit abstractions – a proper partitioning of the heuristic computation into parts that can be shared between search states and parts that must be computed online per state

Databased Implicit Abstractions vs. state-of-the-art

domain	$h^{\mathfrak{F}}$	$h^{\mathfrak{I}}$	$h^{\mathfrak{FI}}$	*MS*-10^4	*MS*-10^5	$\mathrm{HSP}^*_\mathrm{F}$	Gamer	blind	$h_{\max}$
IPC1-5	**368**	337	350	332	285	277	315	296	318

domain	$h^{\mathfrak{F}}$	$h^{\mathfrak{I}}$	$h^{\mathfrak{FI}}$	$\mathrm{HSP}^*_\mathrm{F}$	Gamer	blind
IPC6	**134**	124	126	108	130	117

Contributions

1. Discovering new islands of tractability for both satisficing and cost-optimal planning.
2. Implicit abstraction heuristics for cost-optimal planning.
3. Optimal composition of abstraction heuristics.

Heuristics Composition

Given: a planning problem and a set of admissible heuristics

Find: an admissible heuristic that exploits the heuristics in the set

Solution 1

heuristic that returns maximum over the heuristics in the set

Heuristics Composition

Given: a planning problem and a set of admissible heuristics

Find: an admissible heuristic that exploits the heuristics in the set

Solution 1

heuristic that returns maximum over the heuristics in the set

Solution 2

heuristic that returns sum of the heuristics in the set

♠ in special cases admissible

Admissible Cases

All-in-one/nothing-in-rest

Account for the whole cost of each action in computing a single heuristic in the set, while ignoring the cost of that action in computing all the other heuristics in the set.

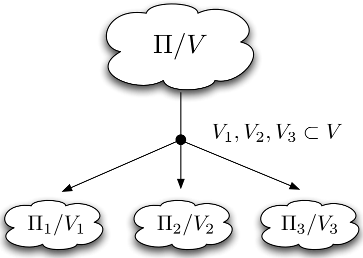
Exploited in Multiple Heuristics

- additive pattern database (PDB) heuristics
- constrained PDB heuristics
- m-reachability heuristics

Action-Cost Partitioning – Basic Idea
ICAPS 2008b, AIJ 2010

Action-Cost Partitioning

For each planning task's action a, if it can possibly be counted by more than one heuristic in the ensemble, then one should ensure that the cumulative counting of the cost of a does not exceed its true cost in the original task.



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

Optimizing Action-Cost Partitioning

Pitfalls

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

$\rightsquigarrow$ *"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)

Abstraction Heuristics – Solution

Procedure:

Given: (i) a planning task Π,
(ii) a state s, and
(iii) a set of admissible heuristics

Find: an optimal action-cost partition for s

- The procedure is fully unsupervised
- The procedure is based on a linear programming formulation of that optimization problem.
- Works for all known to us explicit and implicit abstractions.

Optimal vs. Uniform

	Forks	IForks	Both

Optimal vs. Uniform

	Forks	IForks	Both
☺ Average decrease in expanded nodes	9.55	186.62	43.98

Optimal vs. Uniform

	Forks	IForks	Both
☺ Average decrease in expanded nodes	9.55	186.62	43.98
☹ Average increase in evaluation time	319.34	71.54	354.53

Optimal vs. Uniform

	Forks	IForks	Both
☺ Average decrease in expanded nodes	9.55	186.62	43.98
☹ Average increase in evaluation time	319.34	71.54	354.53

☺ Computing optimal cost partitioning for initial state only improves overall performance

Optimal for Initial State vs. Uniform action-cost partitions

domain ($\mathcal{D}$)	$h^{\mathcal{F}}$		$h^{\mathcal{I}}$		$h^{\mathcal{FI}}$	
	O_I	U	O_I	U	O_I	U
airport-ipc4	22	22	22	20	21	21
blocks-ipc2	21	21	21	18	21	18
depots-ipc3	7	7	7	4	7	7
driverlog-ipc3	12	12	13	12	12	12
freecell-ipc3	5	5	4	4	5	4
grid-ipc1	2	2	2	1	1	1
logistics-ipc2	24	22	21	16	21	16
logistics-ipc1	6	6	5	4	5	5
miconic-strips-ipc2	53	51	53	50	53	50
mprime-ipc1	23	23	23	22	21	21
pipes-notank-ipc4	17	17	18	15	16	16
pipes-tank-ipc4	11	11	11	9	9	9
rovers-ipc5	7	6	7	7	7	6
satellite-ipc4	6	6	7	6	7	6
schedule-strips	49	46	49	40	47	46
zenotravel-ipc3	13	11	11	11	13	11
Total IPC1-5	378	368	371	337	367	350

Follow-Up

- Enriching heuristics with landmark information
  (Domshlak, K, & Lefler, ICAPS 2010)
- Controlling cost partitioning
  (Karpas, K, & Markovitch, ICAPS 2011)
- Satisficing search with admissible heuristics
  (Bahumi, Domshlak, & K, HDIP 2011)

Summary

- New results on complexity of planning
- Formal and empirical results on abstraction-based admissible heuristics
  - from small projections to implicit abstractions
  - optimal combination of multiple abstractions

Future work:

- more tractability results for (cost-optimal) planning
- solving LPs efficiently

- optimization of abstraction selection
- optimization of variable-domains abstraction
- approximation-oriented implicit abstractions
- ...