Operator-Potentials in Symbolic Search: From Forward to Bi-directional Search

Daniel Fišer, Álvaro Torralba, Jörg Hoffmann
danfis@danfis.cz, alto@cs.aau.dk, hoffmann@cs.uni-saarland.de

Potential Heuristics

A **potential function** is a function $\mathtt{P}$ mapping facts to numerical values, $\mathtt{P} : \mathcal{F} \mapsto \mathbb{R}$.
A **potential heuristic** for $\mathtt{P}$ maps each state $s \in \mathcal{R}$ to the sum of potentials of facts in s, i.e.,

$$h^{\mathtt{P}}(s) = \sum_{f \in s} \mathtt{P}(f).$$

The potential function is typically inferred by solving a linear program with a set of constraints ensuring goal-awarness:

$$\sum_{V \in \mathcal{V}} \max_{f \in \mathcal{D}(V)} \mathtt{P}(f) \leq 0,$$

and consistency:

$$\underbrace{\sum_{V \in \text{vars}(\text{eff}(o))} \max_{f \in \mathcal{D}(o,V)} \mathtt{P}(f)}_{\text{potential of deleted facts}} - \underbrace{\sum_{f \in \text{eff}(o)} \mathtt{P}(f)}_{\text{potential of added facts}} \leq \mathrm{c}(o).$$

Operator-Potential Heuristics

Given a potential function $\mathtt{P}$, **operator-potential function** assigns a numerical value to each operator:

$$\mathtt{Q}(o) = \sum_{f \in \text{eff}(o)} \mathtt{P}(f) - \sum_{V \in \text{vars}(\text{eff}(o))} \max_{f \in \mathcal{D}(o,V)} \mathtt{P}(f).$$

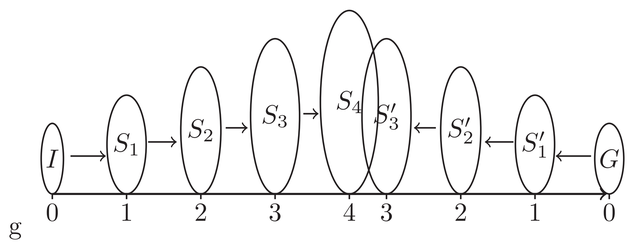
If preconditions on all affected variables are known exactly for all operators, an **operator-potential heuristic** for a given state s reached by a sequence of operators $\langle o_1, \ldots, o_n \rangle$ is defined as:

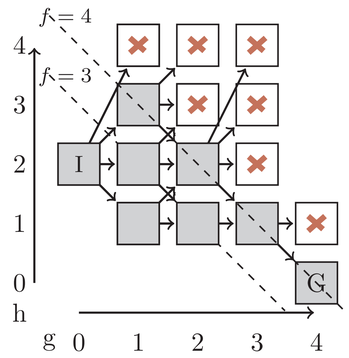
$$h^{\mathtt{Q}}(s) = \sum_{f \in I} \mathtt{P}(f) + \sum_{i \in [n]} \mathtt{Q}(o_i)$$

and then it holds that $h^{\mathtt{Q}}(s) = h^{\mathtt{P}}(s)$ for all reachable states s.

Bi-directional Symbolic Search

Symbolic search utilizes Binary Decision Diagrams (BDDs) to concisely represent sets of states.
With heuristics, sets of states are partitioned by their g-value and h-value which allows to use A*.
Bidirectional: perform a backward and a forward search—generally less effort, as search effort often scales exponentially with search depth





Operator-Potential Heuristics in Backward Direction

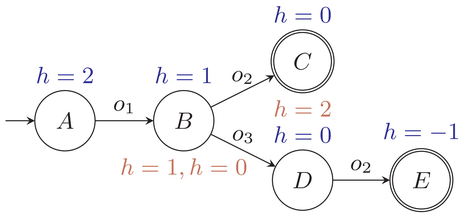
Operator-potentials allow us to partition operators by their cost and the change of the heuristic value they induce (Δh). And this in turn allows us to partition sets of states by both g-value and h-value because for each operator we know how g and h-values of the successor states change.
It turns out, the very same operator-potential heuristics inferred in the forward direction can be used without change in the backward direction as well.

Estimated **init-to-goal** distance $h=2$ $h=1$ $h=2$ $h=2$ $h=1$ $h=0$

$I \xrightarrow{\Delta h=-1} s_1 \xrightarrow{\Delta h=1} s_2 \xrightarrow{\Delta h=0} s_3 \xrightarrow{\Delta h=-1} s_4 \xrightarrow{\Delta h=-1} G$

Estimated **goal-to-init** distance $h=0$ $h=1$ $h=0$ $h=0$ $h=1$ $h=2$

However, the operator-potential heuristics in backward direction may be **inconsistent** whenever there are multiple goal states with different heuristic values.



Experimental Results

	Domain	$\overleftarrow{\mathtt{b}}$	$\overleftarrow{\mathtt{A+I}}$	$\overleftarrow{\mathtt{I}}$	$\overleftarrow{\mathtt{S_{1k}+I}}$	$\overleftarrow{\mathtt{M_2+I}}$
Domains with non-unit cost	agricola (20)	**6**	4	3	2	4
	barman (34)	**13**	6	6	6	6
	data-network (20)	**8**	4	3	4	4
	elevators (50)	**16**	10	10	10	10
	floortile (40)	**34**	31	27	31	31
	parcprinter (50)	**41**	33	35	33	32
	petri-net-align (20)	**16**	1	1	1	1
	sokoban (50)	**43**	38	38	38	38
	woodwork (50)	**49**	36	33	36	36
	pegsol (50)	22	25	**27**	26	25
	scanalyzer (50)	21	**23**	**23**	**23**	**23**
	tetris (17)	6	**11**	10	10	**11**
Domains with unit cost	hiking (20)	**15**	12	11	14	12
	rovers (40)	**13**	12	11	12	12
	airport (50)	23	**25**	**25**	23	23
	blocks (35)	24	23	**32**	22	23
	logistics (63)	19	19	**24**	19	19
	mprime (35)	12	**13**	11	10	**13**
	parking (40)	0	0	**4**	0	0
	pipesw-tank (50)	6	6	6	**7**	6
	driverlog (20)	9	**10**	**10**	**10**	**10**
	freecell (80)	20	21	**29**	21	21
	nomystery (20)	13	14	**17**	13	14
	trucks (30)	10	**12**	**12**	10	**12**
	visitall (40)	16	**19**	18	18	**19**
	.....					
	Σ (1697)	**785**	727	739	714	720

Coverage of variants of symbolic **backward** search
Blind search is superior

	$\emptyset$	$\overleftarrow{\mathtt{b}}$	$\overleftarrow{\mathtt{A+I}}$	$\overleftarrow{\mathtt{I}}$	$\overleftarrow{\mathtt{S_{1k}+I}}$	$\overleftarrow{\mathtt{M_2+I}}$	$\overleftarrow{\text{oracle}}$
$\emptyset$	—	785	727	739	714	720	843
$\overrightarrow{\mathtt{b}}$	936	1 025	950	961	934	951	1 047
$\overrightarrow{\mathtt{A+I}}$	1 109	**1 128**	1 100	1 087	1 093	1 099	1 139
$\overrightarrow{\mathtt{I}}$	996	1 032	985	1 000	987	988	1 058
$\overrightarrow{\mathtt{S_{1k}+I}}$	1 081	1 110	1 073	1 066	1 062	1 074	1 121
$\overrightarrow{\mathtt{M_2+I}}$	1 103	1 120	1 092	1 076	1 085	1 097	1 132
$\overrightarrow{\text{oracle}}$	1 131	1 151	1 111	1 103	1 112	1 114	1 162

Coverage of combinations of forward and backward symbolic search.
Oracle: predicts what is the best forward/backward search per instance