Merging Cartesian Abstractions for Classical Planning

Mauricio Salerno[1], Raquel Fuentetaja[1], David Speck[2], Jendrik Seipp[3]
Universidad Carlos III de Madrid[1], Basel University[2], Linköping University[3]

Cartesian Abstractions in Planning

- **Optimal Classical Planning:** cheapest sequence of actions that achieve the goals
- **A*:** prominent solution method, needs *domain independent* heuristics
- **Abstractions:** solve a simpler version of the problem, use *solution* cost as heuristic
- **Cartesian Abstractions:** prominent abstraction type, built with **CEGAR**

Task Decomposition

- One big abstraction: converges to h^*, suffers from **diminishing returns**
- **One task per goal**, and builds small abstractions
- **Cost partitioning** combines the estimates of the small abstractions **admissibly**

Limitations of each approach:

- **Big abstraction:** considers all goal interactions, expensive to build
- **Small abstractions:** cheap to build, misses goal interactions

New approach: Multi-Goal Cartesian Abstractions

How to select which subsets of goals to build abstractions for?

- Too many alternatives: $\mathbf{2^n - 1}$ subsets for **n** goals
- Can't know in advance what is useful

Systematic approach:

- Start with **single-goal** abstractions
- Build **multi-goal** (first pairs, then triples, etc.) abstractions if resources allow it

Merging Abstractions

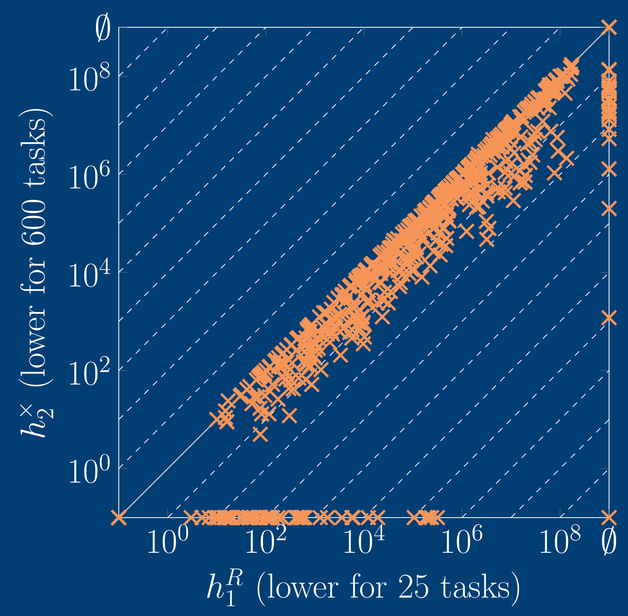
Avoid repeating work by **merging** existing abstractions:

- For **2-goal** abstraction with goals i, j: **merge 1-goal** abstractions for **i** and **j**
- **Merging**: synchronized product of transition systems
- **Efficiency**: avoid re-creating abstractions from scratch

Future Work

- Identify promising goal subsets to build multi-goal abstractions
- Cost partitioning methods tailored for multi-goal abstractions

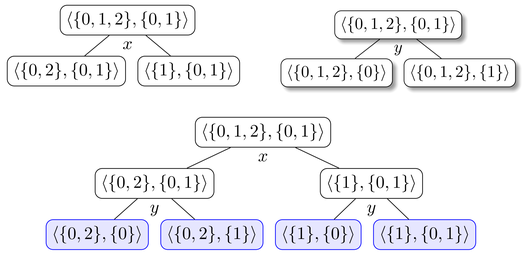
Finding the sweet spot between one big abstraction and many small ones



Check out the paper!

Merge Cartesian Abstractions?

- $\mathcal{V} = \{x, y\}$
- $\mathcal{I} = \{x = 0, y = 0\}$
- $\mathcal{G} = \{x = 1, y = 1\}$
- $dom(x) = \{0, 1, 2\}$
- $dom(y) = \{0, 1\}$



What About Cost Partitioning?

More abstractions can produce **worse heuristics**!
When is it *always* better to have more abstractions?
What if superseded abstractions are **dropped**?

Cost Partitioning	Keep	Drop
Optimal	✓	✓
Saturated Post-hoc	✓	✗
Saturated	✗	✗

More Results!

		Refine					Merge & Refine				+Landmarks	
		$h^\mathcal{G}$	h^R_1	h^R_2	h^R_3	h^R_4	$h^\times_2$	$h^\times_3$	$h^\times_4$	$h^\times_\infty$	h^L_1	h^L_2
Refine	$h^\mathcal{G}$	–	11	7	7	6	6	5	5	5	10	6
	h^R_1	**14**	–	0	1	0	0	0	0	0	1	3
	h^R_2	**20**	**12**	–	1	2	2	3	3	3	9	7
	h^R_3	**22**	**14**	**5**	–	**3**	6	6	6	6	10	10
	h^R_4	**21**	**12**	**6**	**3**	–	4	3	3	3	8	9
M & R	$h^\times_2$	**25**	**19**	**14**	**12**	**11**	–	3	**3**	**3**	**11**	6
	$h^\times_3$	**25**	**20**	**14**	**13**	**12**	**4**	–	**1**	**2**	**12**	8
	$h^\times_4$	**25**	**20**	**14**	**13**	**12**	**3**	0	–	**1**	**12**	8
	$h^\times_\infty$	**24**	**19**	**13**	**12**	**11**	2	0	0	–	**11**	7
+L	h^L_1	**22**	**13**	**11**	**11**	**11**	9	8	8	8	–	2
	h^L_2	**24**	**19**	**16**	**14**	**15**	**9**	**11**	**11**	**11**	**11**	–
	C	844	853	868	875	875	888	889	888	887	1009	1026