Utilising Uncertainty for Efficient Learning of Likely-Admissible Heuristics

ICAPS 2020

Ofir Marom and Benjamin Rosman

University of the Witwatersrand: Rail Lab
South Africa

Heuristic Based Planning

- improve planning performance
- requires domain knowledge
- hard to construct

Machine Learning

2	1	10	3
4	14	6	7
8	9	5	11
12	13	15	

2	12	10	3
14	1	6	7
8	9	4	5
	13	15	11

2	12	1	10
14		6	7
3	8	4	5
9	13	15	11

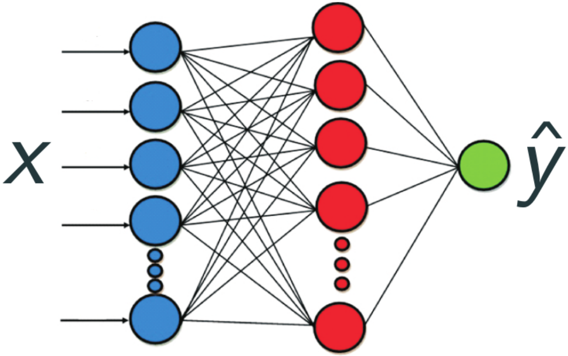
3	12	1	10
2	14	6	7
8	13	4	5
9		15	11

Machine Learning

- $plan = (s_0, s_1, ..., s_n)$
  - $x_i = Extract\_Features(s_i)$
  - $y_i = Cost\_To\_Goal(s_i)$

Machine Learning

Supervised learning: mapping $x \rightarrow \hat{y}$



Does this Scale?

1	6	2	3	4
5		7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24

7.41×10^{11} larger than 15-puzzle

Bootstrap Learning

Initialise (weak) heuristic h_0

Repeat:

- generate 24-puzzle tasks (randomly)
    - try solve each in time-limit with $\hat{y}$
    - if can't solve "enough"
        - increase time-limit
    - else
        - update $h_{i-1} \rightarrow h_i$

Learning Heuristic Functions for Large State Spaces, Arfaee, Zilles and Holte, 2011.

Bootstrap Learning

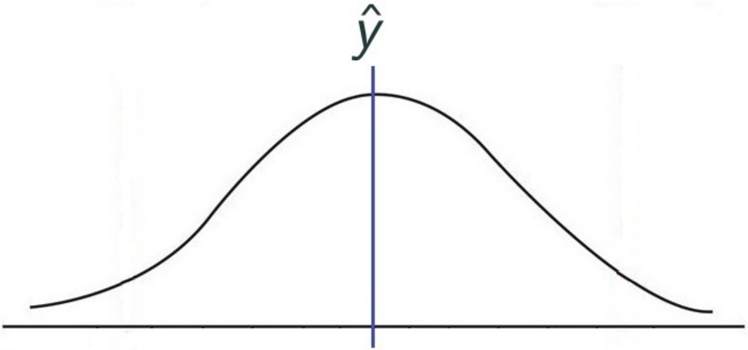
Shortcomings:

- inefficient training
- high suboptimality ($\approx$ 8% for 24-puzzle)

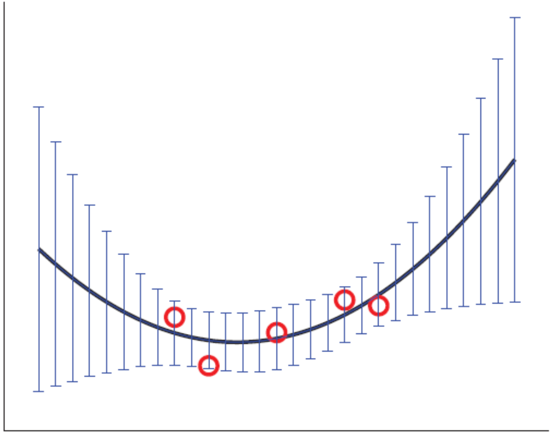
Our contribution addresses these

Bayesian Learning

- standard: $x \to \hat{y}\ (\approx y)$
- Bayesian: $x \to y \sim \mathcal{N}(\hat{y}, \sigma_a^2 + \sigma_e^2)$

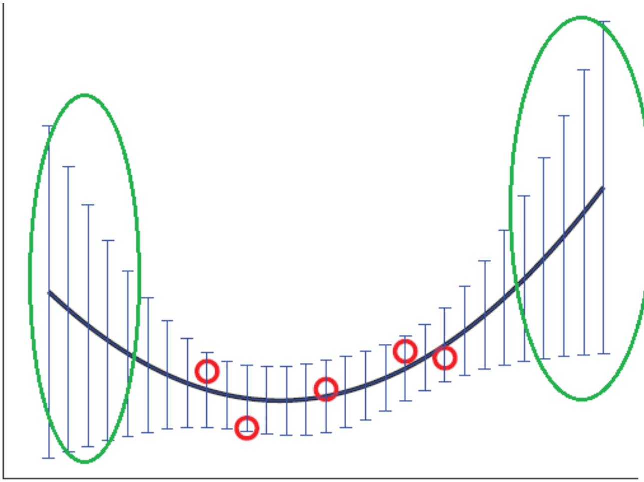


Bayesian Regression

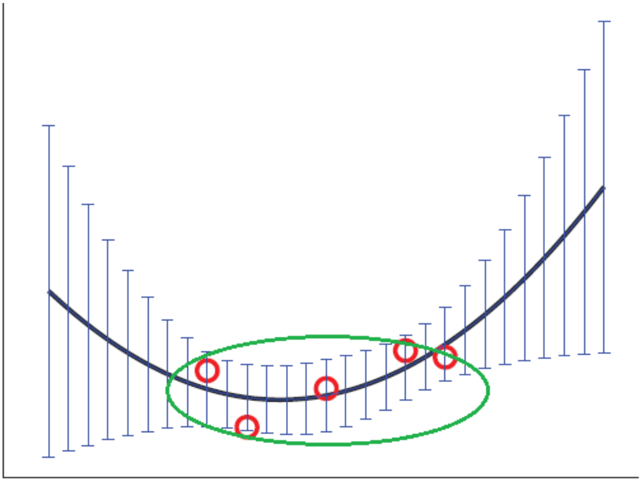


Machine Learning: A Probabilistic Perspective, Murphy, 2012.

σ_e^2: Epistemic Uncertainty



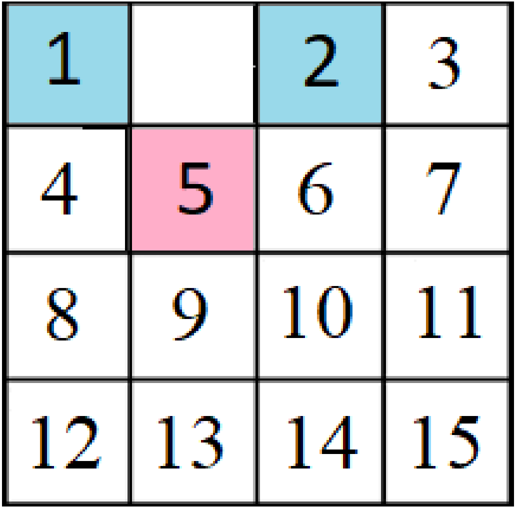
σ_a^2: Aleatoric Uncertainty



Efficient Task Generation

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

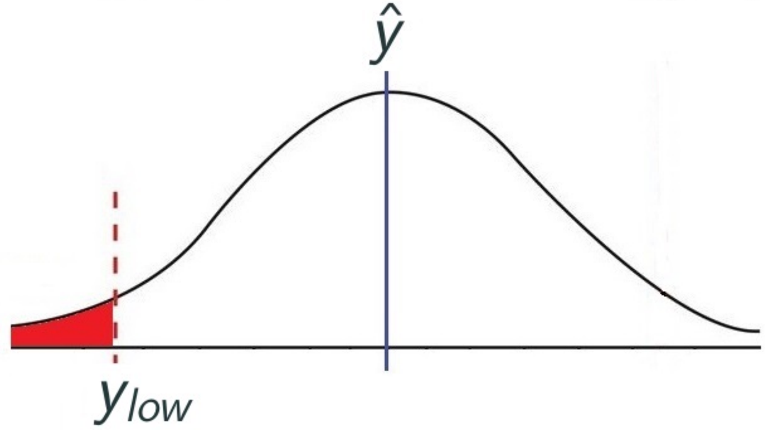
Efficient Task Generation



Efficient Task Generation

1	5	2	3
4		6	7
8	9	10	11
12	13	14	15

Reduce Suboptimality



Likely-Admissible and Sub-Symbolic Heuristics, Ernandes and Gori, 2004.

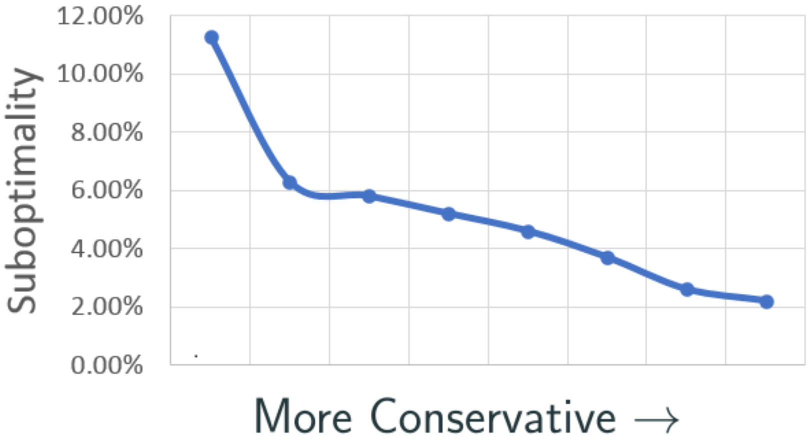
Main Algorithm

Initialise (weak) heuristic h_0

Repeat:

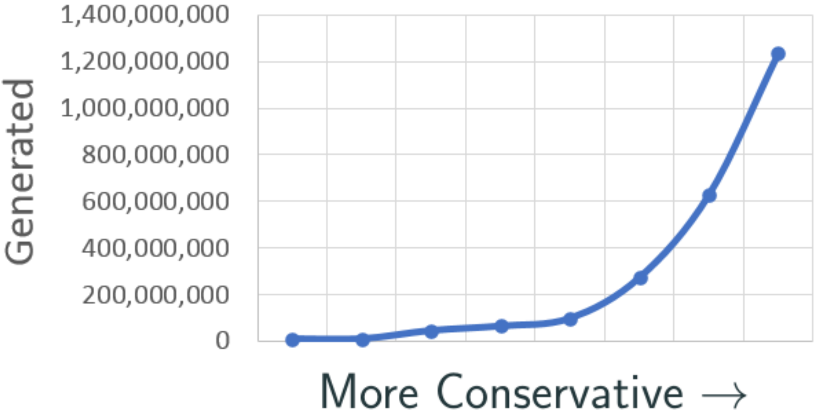
- generate 24-puzzle tasks (efficiently)
    - try solve each in time-limit with y_{low}
    - if can't solve "enough"
        - make y_{low} less conservative
    - update $h_{i-1} \to h_i$

Results



12.00%
10.00%
8.00%
6.00%
4.00%
2.00%
0.00%
Suboptimality
More Conservative →

Results


Generated
1,400,000,000
1,200,000,000
1,000,000,000
800,000,000
600,000,000
400,000,000
200,000,000
0
More Conservative →

Summary

Utilise uncertainty to:

- efficiently generate training tasks
- reduce suboptimality