Reinforcement Learning for Classical Planning

Viewing Heuristics as Dense Reward Generators

Clement Gehring*, Masataro Asai*, Rohan Chitnis, Tom Silver,
Leslie Pack Kaelbling, Shirin Sohrabi, Michael Katz

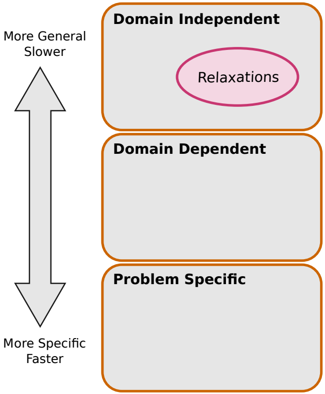
Recorded for the PRL workshop, ICAPS 2021

MIT-IBM Watson AI Lab

LIS LEARNING & INTELLIGENT SYSTEMS

Heuristics are necessary for efficient planning.

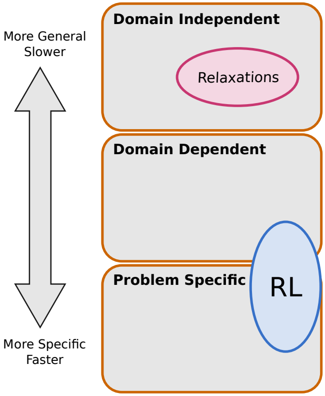
- More specific heuristics can better leverage domain/problem structure.
- Domain dependent heuristics are hard to find.
- Problem specific heuristics have limited applications.



Value-based reinforcement learning (RL):

- can learn **optimal heuristics** (typically problem specific but not always), **but**
- requires **many interactions** and **a lot of computation**.

Our goal: combine RL and domain **independent** heuristics to efficiently learn domain **dependent** heuristics.



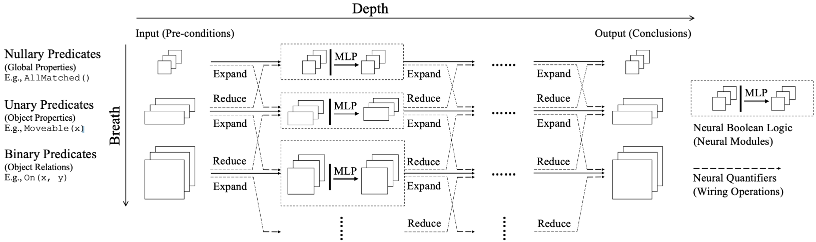
Contributions

Our two main contributions are:

1. using neural logic machines to learn **goal** and **problem** conditioned heuristics, and
2. using potential-based reward shaping to efficiently learn a domain **dependent** heuristic as a correction to a domain **independent** heuristic.

Representing domain dependent heuristics

- Conventional feed-forward neural networks require fixed input size.
- We encode the grounded **state** and **goal** predicates with binary N-d arrays.
- We represent the heuristic with a **neural logic machine**.[1]



[1] Honghua Dong et al. "Neural Logic Machines". In: *ICLR*. 2018.

- We can directly apply non-linear RL methods to learn NLMs.

- RL is terribly slow when rewards are **sparse**.

Solution: add additional rewards, i.e., **reward shaping**

Warning: careless shaping will change your problem in undesirable ways.

"Careful what you wish for!"

Potential-based Reward Shaping

Theoretically nice approach: **potential-based reward shaping**

Define a **new reward function**:[2]

$$\hat{r}(s, a, s') = r(s, a, s') + \gamma\phi(s') - \phi(s)$$

- Preserves optimal policies
- Allows us to inject prior knowledge (e.g., domain independent heuristic)
- "Shaped" value function, $\hat{V}^*_\gamma$, doesn't preserve state ordering
- To plan, we can retrieve the original value function by

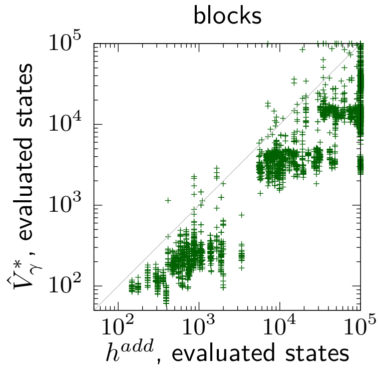
$$V^*_\gamma(s) = \hat{V}^*_\gamma(s) + \phi(s)$$

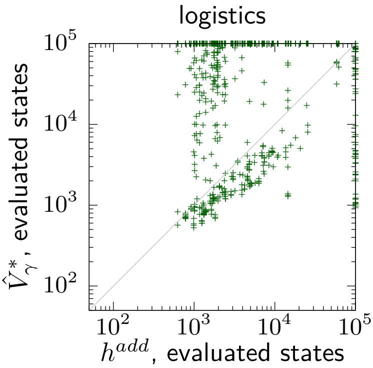
[2] Andrew Y Ng, Daishi Harada, and Stuart J Russell. "Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping". In: *ICML*. 1999.

- We plan with greedy best-first search using the learned heuristic:

$$V^*_\gamma(s) = \hat{V}^*_\gamma(s) + \phi(s)$$

- Train on **small** instances (fast and easy), e.g., 2-6 blocks.
- Evaluate on **unseen** and **larger** instances, e.g., 10-50 blocks.
- Showing **best** and **worst** domains out of **8 domains total**.





- We use RL to learn a domain **dependent** heuristic.
- We **generalize** over problem instances by using **neural logic machines**.
- We leverage domain **independent** heuristic to accelerate learning using potential-based **reward shaping**.
- **Potential-based reward shaping** allows us to learn **corrections** to a classical heuristic, enabling us to plan.
- Our method is capable of learning **goal** and **problem** conditioned heuristics capable of generalizing to **larger** instances.

Thank you for watching!